proven extremely effective. Extremely stable output pressures are achieved by recirculating the controlled pressure to the valve element, making it possible to maintain stable output pressures in the face of interference factors like downstream leakage, supply-pressure fluctuations, and viscosity variations in the hydraulic fluid. Figure 10.33 shows a typical version of the three-way pressure-control valve, the spool valve. This design provides flow rates of 4 dm^3/min at pressure differentials of 2 bar. In order to limit the effects of hysteresis on the pressure/flow response characteristic, mechanical friction must be minimized and materials with low coercive field force must be employed and heat-treating processes are also needed to minimize material-related hysteresis. Choppers in the current-controlled end stage reduce power loss and induce friction-reducing micromovement in the armature. Pressure regulators are also available as seat valves.

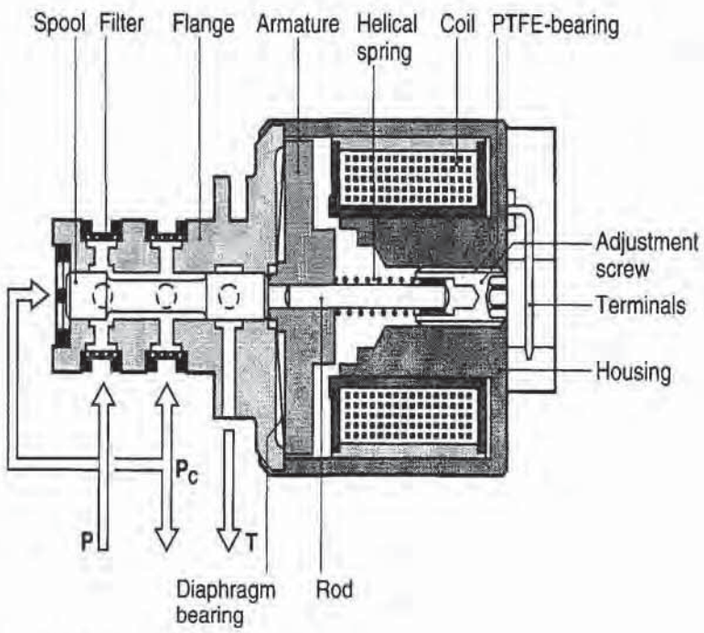


FIGURE 10.33 Variable-pressure solenoid.

PWM Solenoids (Pulse-Width Modulated Solenoid Valves). On/off poppet valves are particularly well suited for direct actuation from microcomputers. The component layout is less complicated than that found in proportional valves, allowing reductions in manufacturing costs. Because microcomputers operate on the basis of discrete time, equidistant setpoint selection, direct valve actuation via constant-frequency pulse represents the optimum design configuration.

When the solenoid-actuated valve is switched on and off at a constant supply pressure, the resulting average outlet pressure is a function of the ratio between open and closed states. However, this design lacks the proportional valve's ability to compensate for interference due to such factors as system leakage, temperature, and fluctuations in supply pressure. This type of actuator thus represents a reasonably priced alternative for applications in which the control pressure is not the controlled variable, but rather is used as a manipulated variable.

These units can be used in automatic transmissions to select gear ratios, or to reduce torque-converter slip losses by modulating the contact pressure of the lockup clutch. Simple switching output stages or peak-and-hold circuits govern the electric control signal to the PWM valve. When used in conjunction with the appropriate valve design, this type of control can be used to obtain cycle frequencies in excess of 100 Hz. In some applications, such as continiously variable transmissions (CVT), a life expectancy of 2×10^9 cycles is demanded.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

10.3.6 Actuators for Headlight Vertical Aim Control

Devices enabling adjustment of headlight range enhance safety by maintaining the correct aim under all vehicle-load conditions. Range adjustment can be manual, with a driver-operated switch, or it can be automatic. The same headlight-adjustment actuators are employed for both designs. The actuator is mounted directly on the headlamp bracket or housing, and adjusts the headlight insert or the reflector.

The actuators are dc or step motors producing a rotary motion which gear-drive units then convert to linear movement. Due to the fixed relationship between the motor's incremental response (steps) and the attendant linear motion, no travel sensor is required with step motors. The total adjustment range extends as far as 8 mm. A single step corresponds to a turning angle of 15 degrees, or an adjustment travel of 0.03 mm. The motor returns to the initial (zero) position (mechanical travel limiter) each time the system is activated in order to prevent step losses.

The step motor combines the following advantages:

- Extremely precise positioning through digital control.
- High adjustment speeds of up to 8 mm/s; different control frequencies can be selected to obtain variations. This system allows graduated adjustment at constant vehicle speeds (consistent lighting) and rapid response during acceleration and braking.

10.4 TECHNOLOGY FOR FUTURE APPLICATION

The motivation to develop new actuators is created by the potential advantages of new manufacturing and driving techniques in combination with new materials. The following examples illustrate representative fields of innovation.

10.4.1 Micromechanical Valves

Micromechanics technology stems from the adaptation of production methods employed in microelectronics. Lithographic miniaturization procedures and etching and assembly techniques make it possible to manufacture minute structures with a high degree of precision. Micromechanical production methods have already become established in the field of sensor manufacture. Electronic circuitry and sensors can be manufactured from the same material (primarily silicon) in simultaneous production processes to furnish integrated components; the benefits associated with this process represent numerous potential advantages.

In automotive applications, microactuators display potential as elements in open-loop control systems where low control power is to be converted, for instance, regulating the flow of fluids in hydraulic or pneumatic systems and metering fuel.

For micromechanical manufacturing, valve designs are required in which planar units with relatively low structure heights are stacked. The energy conversion principles employed to operate the valves correspond to the planar structure (e.g., electrostatic drive, electrothermal or piezoelectric flux converters).

The pressure-balanced seat valve is a basic example of this kind of microvalve (Fig. 10.34). This type of microvalve is produced in a manufacturing process entailing sequential structuring and bonding of four wafers. Depending on the required dimensions for the single valve and the resulting surface requirement, as many as several hundred elements can be produced simultaneously on each wafer stack. Further efforts must be directed to solving packaging problems before these microstructures can be used to control fluid supply and discharge processes.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

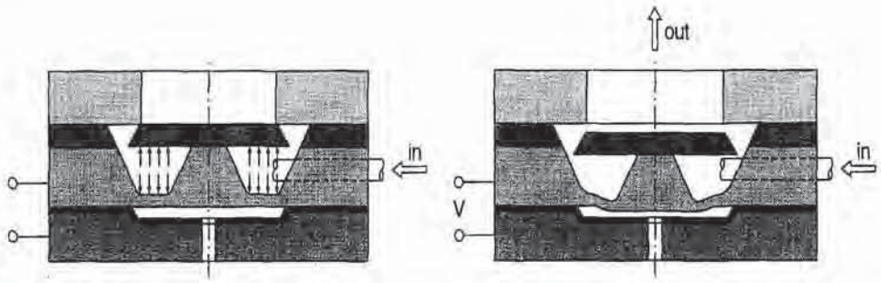


FIGURE 10.34 Pressure-compensated seat valve.

10.4.2 Positive-Engagement Friction Drives

Single piezoelectric actuators provide high dynamic positioning response over short actuating distances. The search for suitable mechanisms employing active piezoceramics to extend the effective travel range has given rise to a multiplicity of drive concepts sharing common attributes:

- Combination of single high-frequency motions (up to the ultrasonic range) to achieve a continuous drive movement
- Output element actuation using friction engagement

Ultrasonic Motor. Of all the designs utilizing piezoelectric actuator technology, the ultrasonic traveling wave device has reached the most advanced stage of development (Fig. 10.35). A stator ring featuring teeth on the upper surface, and made of a material providing low material damping (bronze), is flex-mounted on an end shield via a central diaphragm. A flat piezoceramic ring is bonded to the underside of the stator ring. This piezoring is polarized along the vertical axis of the ring plane. The direction of the polarization changes segmentally (Fig. 10.36).

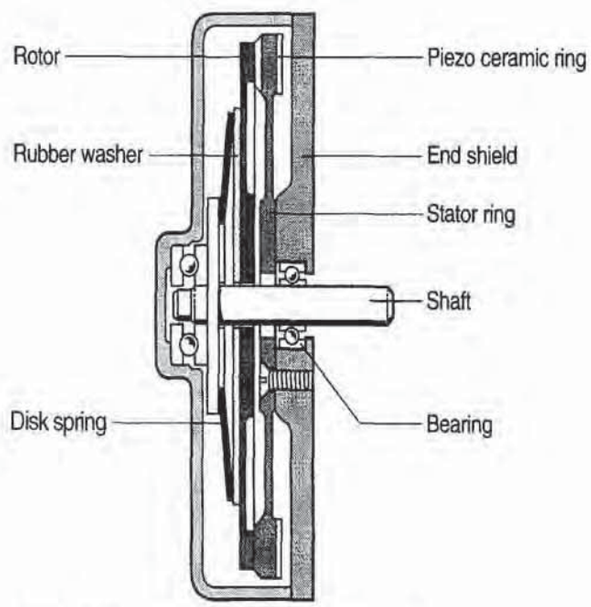


FIGURE 10.35 Traveling-wave motor, Shinsei type.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

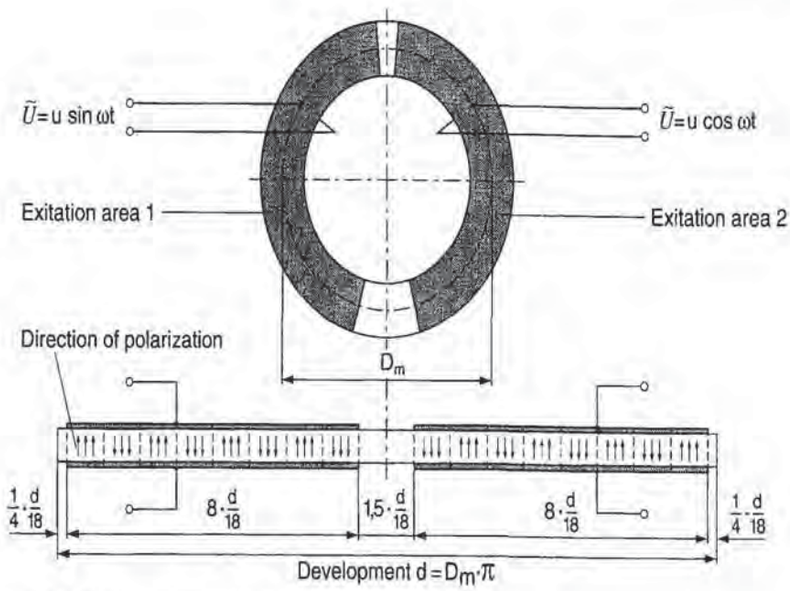


FIGURE 10.36 Segmentation and control of the piezoring.

The electrical contacts are arranged to provide two excitation regions, each featuring eight segments and two small zones. The zone length corresponds to either ½ or 1½ segments. Thus a segment of the excitation zone extends through ⅛ of the stator's circumference. When voltage is applied to an excitation region, the piezos of one polarization direction contract along the circumference direction (lateral piezo effect), while the piezos of the opposite polarization orientation expand. This effect produces a wave-shaped flux in the stator ring in the affected region.

If the region is then excited with ac voltage (for example, 100 V at approximately 40 kHz in existing motors), the result is an oscillation pattern which extends to encompass the entire stator ring. Excitation in the frequency of the ninth flex mode of the stator provokes a standing wave on the entire stator. Resonance step-up is employed to achieve amplitudes of 20 μm.

If an excitation current is now applied at the second excitation region with ac voltage at a 90° lateral displacement, the two waves overlap to form a traveling wave.

In addition to the up and down motion, one point on the upper surface of the stator produces—by means of tilting movement of those teeth which are situated above the neutral axis—a motion along the periphery (±2 μm). When the two motions combine, the surface points move in an elliptical pattern (Fig. 10.37).

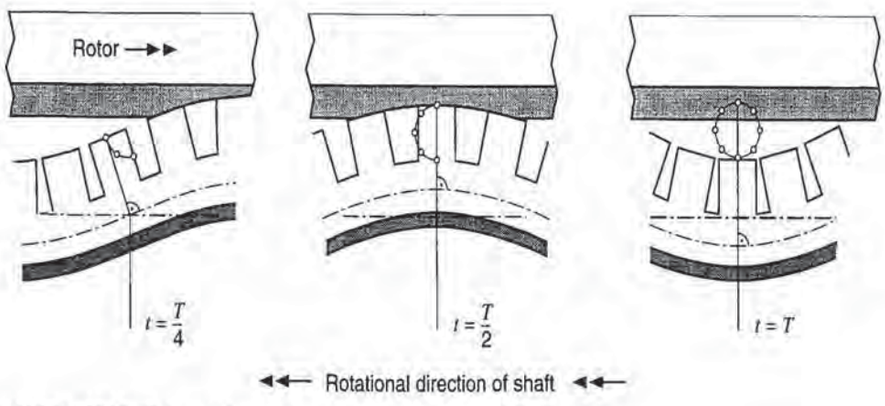


FIGURE 10.37 Drive mechanism.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

A disc spring and a rubber washer are pressing the disk-shaped rotor in Fig. 10.35 against the stator. The elliptical motion of the stator teeth drives the rotor through friction contact with its contact layer. Speed is adjusted by, for example, varying the excitation frequency, with the stator's oscillation amplitude being monitored.

The salient characteristics of the ultrasonic motor are: high torque (up to 1.3 Nm), low rpm range (30 . . . 130 rpm), bidirectional operation, substantial characteristic retension force, free of clearance, no run-on, potentially precise positioning, noiseless operation, and flat unit configuration.

Ultrasonic motors are suitable for use as direct-drive devices. The disadvantages include the substantial, high-frequency control voltage, and the limitations on service life due to wear on the friction surface.

ACKNOWLEDGMENTS

The author wishes to acknowledge the contributions of D. Baumann, W. Brehm, O. Engfer, G. Genter, H. Gnuschke, U. Hafner, G. Hartz, Prof. E. Kallenbach, Technische Universität Ilmenau; T. Kamitsis, Morton International Inc., Odgen Utah; G. Keuper, H. M. Streib, U. Zillgitt. Unless otherwise stated, the collaborators are employees of Robert Bosch GmbH, Stuttgart.

The author acknowledges with gratitude the support of theoretical revision by Dietmar Baumann and the coordination done by Mr. Ortwin Engfer, as well as the typing of the manuscript by Dorothee Ludmann.

GLOSSARY

Actuator The part of an open-loop or closed-loop control system which connects the electronic control unit with the process. The actuator consists of a transformer and a final-control element. Electric positioning signals are converted to mechanical output.

Closed-loop control A process by which a variable is continuously measured, compared with a reference variable, and changes as a result of this comparison in such a manner that the deviation from the reference variable is reduced. The purpose of closed-loop control is to bring the value of the output variable as close as possible to the value specified by the reference variable in spite of disturbances. In contrast to open-loop control, a closed-loop control system acts to offset the effect of all disturbances.

Commutator The commutator is a current switcher. For dc machines, the commutator switches the armature windings so that the resultant force always acts in the same rotary direction. This requires a reversal of the armature winding connection every 180°. The current supply to the armature is via brushes which contact the commutator.

Eddy current In metals moving in an inhomogeneous magnetic field or located in a changing magnetic field, induced currents circulate throughout the volume. Because of their general circulatory nature, these currents are referred to as eddy currents.

Ferrite magnet In small transformers where eddy current losses must be kept to a minimum, the cores are made of ferrites which are complex oxides of iron and other metals. These materials are ferromagnetic, but have relatively high resistivity.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Final-control element The second or last stage of an actuator to control mechanical output.

Numeric field calculation A method of numerically calculating fields with the help of computer programs, such as the methods of finite differences or finite elements.

Open-loop control A process within a system in which one or more input variables act on output variables based on the inherent characteristics of the system. An open control loop is a series of elements that act on one another as links in a chain. In an open control loop, only disturbances that are measured by the control unit can be addressed. The open loop has no effect on other disturbances.

Peripheral magnetic voltage The line integral of the magnetic field strength around a closed path.

$$V_m = \oint \mathbf{H} \cdot d\mathbf{s}$$

According to Ampere's law, the peripheral magnetic voltage is equal to the magnetomotive force Θ.

Piezoelectric effect The direct piezoelectric effect is the ability of a piezoelectric crystal to produce an electric voltage when subjected to a force. The inverse piezoelectric effect is the ability of a piezoelectric crystal to deform when subjected to an electric voltage.

Pilot-controlled actuator An actuator that uses one or more additional energy sources to transform the input signal to an output signal. The pilot-controlled actuator consists of a chain of energy positioners and energy transformers that produces an amplification due to their series configuration.

Self-induction Every current is surrounded by a magnetic field with field lines that are interlinked with the current lines. This leads to an induced source voltage in the conductor or coil when the current strength is changed. This phenomenon is called self-induction.

Weiss' domains Ferromagnetic materials have strong interaction among the atoms. Spontaneous local magnetization can occur, even though no external magnetic field is present. These so-called Weiss' domains, sized from 0.01 to 1 mm, are small elementary magnets, which are randomly distributed in the material and are first directed when an external field is applied.

BIBLIOGRAPHY

Actuators Basics

"DIN 19226 Regelungstechnik und Steuerungstechnik: Begriffe und Benennungen."

Janocha, H., *Aktoren: Grundlagen und Anwendungen,* Springer-Verlag, Berlin, Heidelberg, New York, 1992.

Kupfmüller, K., *Einführung in die theoretische Elektrotechnik,* Springer-Verlag, 1973.

Pregla, R., *Grundlagen der Elektrotechnik,* Teil I u. II, Hüthig-Verlag, Heidelberg, 1990.

Raab, U., "Modellgestützte digitale Regelung und Überwachung von Kraftfahrzeugaktuatoren," Reihe 8: Meβ-, Steuerungs- und Regelungstechnik Nr. 313, VDI-Verlag, Düsseldorf 1993.

Robert Bosch GmbH, *Automotive Handbook,* 3d ed., 1993.

Robert Bosch GmbH, "Elektronik und Mikrocomputer," 1987.

VDI/VDE-Technologiezentrum Informationstechnik GmbH, "Neue Aktoren," Fachbeilage Mikroperipherik, 1990.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

dc Solenoids

Aldefeld, B., "Numerical Calculation of Electromagnetic Acutators," *Archiv für Electrotechnik,* Bd. 61, 1979, pp. 347–352.

Hickmann, W., "Ein Beitrag zur Rechnerunterstützten Auslegung und Optimierung von Gleichstrommagnetsystemen," Dissertation, TH Darmstadt, 1984.

Kallenbach, E., *Der Gleichstrommagnet,* Akademische Verlagsgesellschaft Geest & Portig KG, Leipzig, 1969.

Kallenbach, E., Bögelsack, G., *Gerätetechnische Antriebe,* Carl Hanser-Verlag, München, Wien, 1991.

Müller, W., "Numerical solution of 2- or 3-dimensional nonlinear field problems by means of the computer program PROFI," *Archiv für Elektrotechnik 6,* 1982, p. 299–307.

Roters, H. C., *Electromagnetic Devices,* John Wiley & Sons, New York, 1967.

Rüdenberg, *Elektrische Schaltvorgänge,* Springer, 1974.

Seely, S., *Electromechanical Energy Conversion,* McGraw-Hill New York, 1962. (Moskau: Energija, 1968).

Step Motors

Kuo, B. C., *Incremental Motion Control—Step Motors and Control Systems,* SRL Publishing Company, Champaign Ill., 1979.

Kuo, B. C., *Theory and Applications of Stepmotors,* West Publishing Co., New York, 1974.

Miller, T. J. E., *Brushless Permanent Magnet and Reluctance Motor Drives,* Clarendon Press, Oxford, 1989.

Takashi, Kenjo, *Stepping Motors and Their Microprocessor Controls,* Clarendon Press, Oxford, 1985.

dc Motors

Moeller, W. *Leitfaden der Electrotechnik,* Band II, Teil I, -Gleichstrommaschinen, Teubner, Stuttgart, 1979.

Ruschmeyer, K., *Motoren und Generatoren mit Dauermagneten,* Expert-Verlag, Grafenau, 1983, ISBN 3-88508-914-9.

Schüler, K., and K. Bringmann, *Dauermagnete,* Springer, Berlin, 1970.

Magnetostrictive Actuators

Clark, A. E., "Ferromagnetiv Materials," Chap. 7, Ed. E.P. Wohlfarth, North-Holland, 1980, pp. 531–589.

Dyberg, J., "Magnetostrictive Rods in Mechanical Applications," *Proc. 1st International Conference on Giant Magnetostrictive Alloys and Their Impact on Actuator and Sensor Technology,* Marbella, Spain, 1986.

Edge Technologies Inc., *TERFENOL-D Notes,* Vol. 3, No. 1, 1990.

Fahlander, M., and M. Richardson, "New Material for the Rapid Conversion of Electric Energy to Mechanical Motion," Feredyn AB, Uppsala, Sweden, 1988.

Janocha, H., and J. Schäfer, "Design Rules for Magnetostrictive Actuators," *Proc. Actuators 92,* VDI/VDE-Technologiezentrum Informationstechnik GmbH, Berlin, 1992.

Kvarnsjo, L., "Principles and Tools for Design of Magnetomechanical Devices Based on Giant Magnetostrictive Materials," Royal Institute of Technologie, Dept. of Plant Engineering, S-10044, Stockholm, Sweden, 1990.

Piezo Actuators

Galvagni, J., and B. Rawal, *Multilayer Electroactive Ceramic Actuators,* AVX Corporation, Myrtle Beach, 1991.

Janocha, H., and D. J. Jendritza, "Piezoaktuatoren—Möglichkeiten und Grenzen einer innovativen Stellgliedtechnologie," *VDI Berichte* Nr. 960, 1992.

Janocha, H., and D. J. Jendritza, "Piezoelektrische Aktoren praxisgerecht einsetzens," *Design & Elektronik 22;* Magna MediaVerlag, Haar bei München, 1992.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

VDI/VDE-Technologiezentrum Informationstechnik GmbH, Piezokeramische Aktoren, Fachbeilage Mikroperipherik me Bd. 5, Heft 1, 1991.

Waanders, J. W., *Piezoelectric Ceramics: Properties and Applications,* Philips Components, 1991.

Electrostatic Actuators

Bart, S. E., T. A. Lober, R. T. Howe, J. H. Lang, and M. F. Schlecht, "Design considerations for microfabricated electric actuators," *Sensors and Actuators,* 14, 1988, pp. 269–292.

Price, R. H., J. E. Wood, and S. C. Jacobsen, "The modelling of electrostatic forces in small electrostatic actuators," *IEEE Solid-State Sensor and Actuator Workshop,* Hilton Head Island, S.C., 1988.

Trimmer, W. S., and K. J. Gabriel, "Design Considerations for a Practical Electrostatic Micromotor," *Sensors and Actuators,* 11, 1987, pp. 189–206.

Electrorheological Fluids

Block, H., and J. P. Kelly, "Electro-rheology—A Review Article," *J Phys. D. Appl. Phys,* 21, 1988, pp. 1661–1667.

Bonnecaze, R. T., and J. F. Brady, "Yield stress in electrorheological fluids," *J. Rheol.,* Vol. 36, No. 1, 1992, pp. 73–115.

Simmonds, A. J., "Electro-rheological valves in a hydraulic circuit," *IEE Proc-D,* Vol. 138, No. 4, 1991.

Stangroom, J. E., "Electrorheological Fluids," *J Physics Technology,* 14, 1983, pp. 290–296.

Memory Alloys

Brinson, L. C., "One Dimensional Constitutive Behavior of Shape Memory Alloys: Thermomechanical Derivation with Noncourtant Material Functions and Redefined Martensite Internal Variable," submitted for publication in *Journal of Intelligent Material Systems and Structures,* 1991.

Duerig, T. W., "Applications of shape memory," *Material Science Forum,* 56–58, 1990, pp. 679–692.

Duerig, T. W., K. N. Melton, D. Stoeckel, and C. M. Wayman, *Engineering Aspects of Shape Memory Alloys,* Butterworth-Heinemann, 1990.

Golestaneh, A., "Shape-memory phenomena," *Phys. Today,* Apr. 1984, pp. 62–70.

Tuominen, S. M., and R. J. Biermann, "Shape Memory Wires," *J. Metals,* 1988, p. 32.

Automotive Actuators for ABS/ASR-Systems

Huber, W., B. Lieberoth-Leden, W. Maisch, and A. Reppich, "New Approaches to Electronic Throttle Control," SAE 910085.

Maisch, W., W. D. Jonner, R. Mergenthaler, and A. Sigl, "ABS5 and ASR5: The New ABS/ASR Family to Optimize Directional Stability and Traction," SAE 930505.

Fuel Injection for Spark-Ignition Engines

Fuel Metering (Spark Ignition Engines), SAE 891832, Nov. 89.

Robert Bosch GmbH, *Automotive Electric/Electronic Systems,* SAE ISBN 0-89883-509-7, VDI ISBN 3-18-419110-9.

Fuel Injection for Diesel Engines

Fischer, W., W. Fuchs, H. Laufer, and U. Reuter, "Solenoid-Valve Controlled Diesel Distributor Injection Pump," SAE 930327.

Franke, G., B. G. Barker, and C. T. Timus, "Electronic Unit Injectors," SAE 885013.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Lauvin, P., A. Löffler, A. Schmitt, W. Zimmermann, and W. Fuchs, "Electronically Controlled High Pressure Unit Injector System for Diesel Engines," SAE 911819.

Mardell, J. E., and R. K. Cross, "An Integrated, Full Authority, Electrohydraulic Engine Valve and Diesel Fuel Injection System, SAE 880602.

Electronic Transmission Control

Brehm, W., and K. Neuffer, "Fast switching PWM-solenoid for automatic transmissions," *Proc. Seventh International Conference on Automotive Electronics,* London, Oct. 9–12, 1989, C 391/046 I Mech E, 1989.

Henry, James P., and David S. Dennis, "Predicting Solenoid Transient Performance," SAE Paper 870473.

Robinson, G., "A Practical Approach to Automatic Transmission Reliability," SAE Paper 910640.

U.S. Patent 4,535,816, "Pressure controller," 1985.

U.S. Patent 4,577,143, "Method and apparatus to convert an electrical value into a mechanical position by using an electromagnetic element subject to hysteresis," 1986.

Technology for Future Application

Fröschle, A., "Analyse eines Piezo-Wanderwellenmotors," Dissertation, Universität Stuttgart, 1992.

Huff, M. A., M. Mettner, A. A. Lober, and M. A. Schmidt, "A Pressure-Balanced Electrostatically-Actuated Microvalve," *Technical Digest, IEEE Solid-State Sensor and Actuator Workshop,* Hilton Head, S.C., June 4–9, 1990, p. 123.

Mettner, M., M. A. Huff, T. A. Lober, and M. A. Schmidt, "How to Design a Microvalve for High Pressure Application," *MME Micromechanics Europe '90,* Berlin, Nov. 26–27, 1990, p. 108.

ABOUT THE AUTHOR

Klaus Müller, after studying mechanical engineering, was employed at the University of Karlsruhe, Germany, investigating the field dynamic behavior of air conditioning system components. In 1976, he joined Robert Bosch GmbH, Stuttgart, where he began his work in advanced engineering of planar oxygen sensors, combustion sensors, and signal evaluation. This was followed by product development of hydraulic units for antilock braking systems and traction control.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

P · A · R · T · 3

CONTROL SYSTEMS

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

CHAPTER 11
AUTOMOTIVE MICROCONTROLLERS

David S. Boehmer
Senior Applications Engineer
Intel Corporation

ENGR & MATH SCI LIBRARY UCLA

A microcontroller can be found at the heart of almost any automotive electronic control module or ECU in production today. Automotive systems such as antilock braking control (ABS), engine control, navigation, and vehicle dynamics all incorporate at least one microcontroller within their ECU to perform necessary control functions. Understanding the various features and offerings of microcontrollers that are available on the market today is important when making a selection for an application. This chapter is intended to provide a look at various microcontroller features and provide some insight into their characteristics from an automotive application point of view.

11.1 MICROCONTROLLER ARCHITECTURE AND PERFORMANCE CHARACTERISTICS

A microcontroller can essentially be thought of as a single-chip computer system and is often referred to as a single-chip microcomputer. It detects and processes input signals, and responds by asserting output signals to the rest of the ECU. Fabricated upon this highly integrated, single piece of silicon are all of the features necessary to perform embedded control functions. Microcontrollers are fabricated by many manufacturers and are offered in just about any imaginable mix of memory, I/O, and peripheral sets. The user customizes the operation of the microcontroller by programming it with his or her own unique program. The program configures the microcontroller to detect external events, manipulate the collected data, and respond with appropriate output. The user's program is commonly referred to as code and typically resides on-chip in either ROM or EPROM. In some cases where an excessive amount of code space is required, memory may exist off-chip on a separate piece of silicon. After power-up, a microcontroller executes the user's code and performs the desired embedded control function.

Microcontrollers differ from microprocessors in several ways. Microcontrollers can be thought of as a complete microcomputer on a chip that integrates a CPU with memory and various peripherals such as analog-to-digital converters (A/D), serial communication units (SIO, SSIO), high-speed input and output units (HSIO, EPA, PWM), timer/counter units, and

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

standard low-speed input/output ports (LSIO). Microcontrollers are designed to be embedded within event-driven control applications and generally have all necessary peripherals integrated onto the same piece of silicon. Microcontrollers are utilized in applications ranging from automotive ABS to household appliances in which the microcontroller's function is predefined and limited user interface is required.

Microprocessors, on the other hand, typically require external peripheral devices to perform their intended function and are not suited to be utilized in single-chip designs. Microprocessors basically consist of a CPU with register arrays and interrupt handlers. Peripherals such as A/D and HSIO are rarely integrated onto microprocessor silicon. Microprocessors are designed to process large quantities of data and have the capability to handle large amounts of external memory. Although microprocessors are typically utilized in applications which are much more human-interface and I/O intensive such as personal computers and office workstations, they are beginning to find their way into embedded applications.

Choosing a microcontroller for an application is a process that takes careful investigation and thought. Items such as memory size, frequency, bus size, I/O requirements, and temperature range are all basic requirements that must be considered when choosing a microcontroller. The microcontroller family must possess the performance capability necessary to successfully accomplish the intended task. The family should also provide a memory, I/O, and frequency growth path that allows easy upgradability to meet market demands. Additionally, the microcontroller must meet the application's thermal requirements in order to guarantee functionality over the intended operating temperature range. Items such as these must all be considered when choosing a microcontroller for an automotive application.

11.1.1 Block Diagram

Usually the first item a designer will see when opening a microcontroller data book or data sheet is a block diagram. A block diagram provides a high-level pictorial representation of a microcontroller and depicts the various peripherals, I/O, and memory functions the microcontroller has to offer. The block diagram gives the designer a quick indication if the particular microcontroller will meet the basic memory, I/O, and peripheral needs of their application. Figure 11.1 shows a block diagram for a state-of-the-art microcontroller. It depicts 32 Kbytes

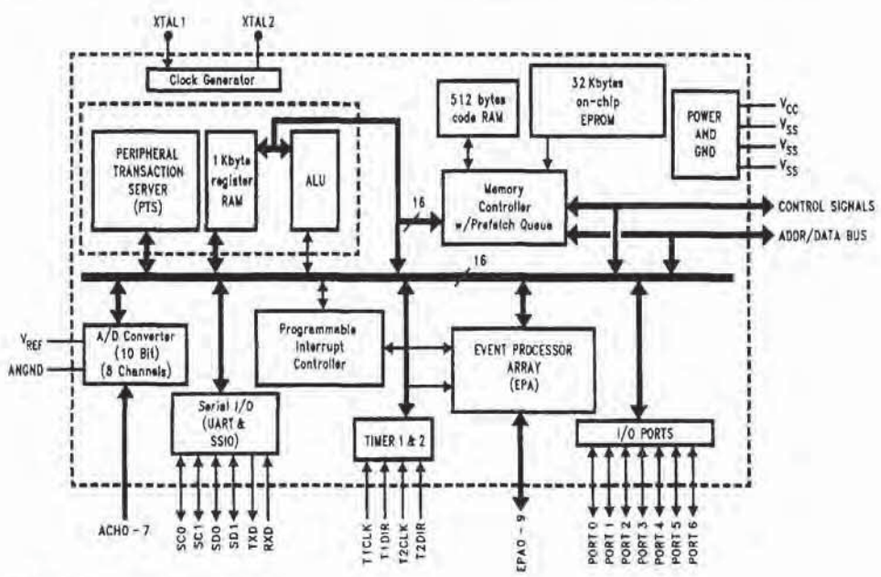


FIGURE 11.1 Microcontroller block diagram.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

of EPROM, 1 Kbyte of register RAM, 6 I/O ports, an A-to-D converter, 2 timers, high-speed input/output (I/O) channels, as well as many other peripherals. These features may be "excessive" to a designer looking for a microcontroller to implement in an automotive trip-computer application but would be excellently suited for automotive ABS/traction control or engine control.

11.1.2 Pin-Out Diagram

A microcontroller's pin-out diagram is used to specify the functions assigned to pins relative to their position on a given package. An example pin-out diagram is shown in Fig. 11.2. Note that most pins have multiple functions assigned to them. Pins that can support more than one function are referred to as multifunction pins. The default function for multifunction pins is normally that of low-speed input and output (discussed later in this chapter). If the user should wish to select the secondary or special function associated with the pin, he or she can do so by writing to the appropriate special function register. There are some exceptions. A good example is pins used for interfacing to external memory. If the device is instructed to power-up executing from external memory as opposed to on-chip memory, the address data bus and associated control pins will revert to their special function as opposed to low-speed I/O.

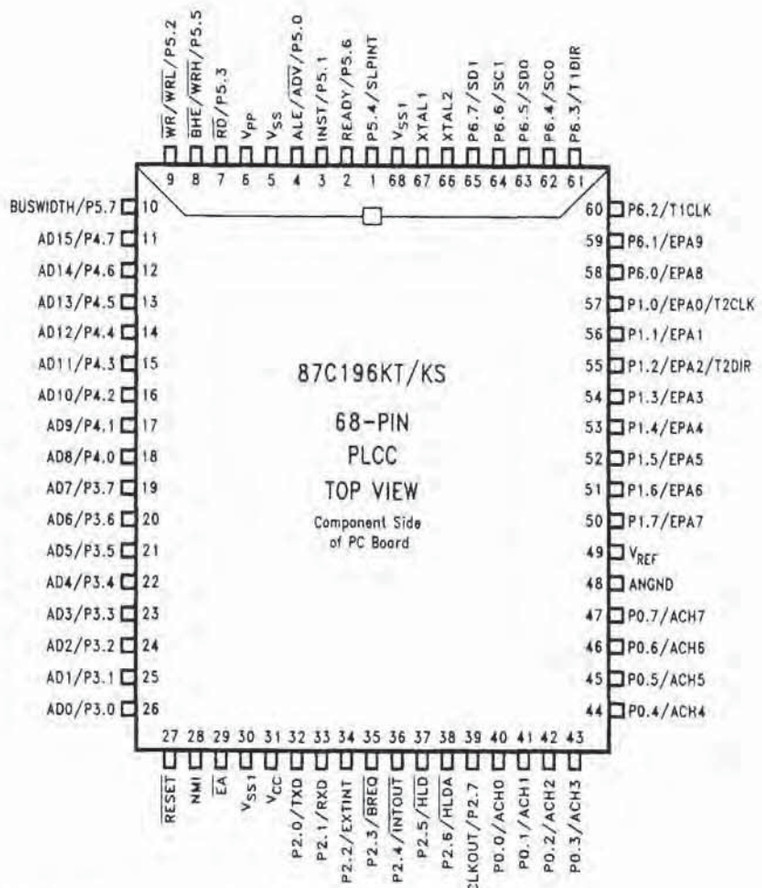


FIGURE 11.2 Microcontroller pin-out diagram.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

11.1.3 Central Processing Unit

The central processing unit or CPU can be thought of as the brain of a microcontroller. The CPU is the circuitry within a microcontroller where instructions are executed and decisions are made. Mathematical calculations, data processing, and control signal generation all take place within the CPU. Major components of the CPU include the arithmetic logic unit (ALU), register file, instruction register, and a microcode engine. The CPU is connected to the bus controller and other peripherals via a bidirectional data bus.

Microcontrollers are, for the most part, digital devices. As digital devices, microcontrollers utilize a binary numbering system with a base of 2. Binary data digits or *bits* are expressed as either a logic "1" (boolean value of true) or a logic "0" (boolean value of false). In a 5-V system, a logic "1" may be simply defined as a +5-V state and a logic "0" may be defined as a 0-V state. A bit is a single memory or register location that can contain either a logic "1" or a logic "0" state. Bits of data can be arranged as a *nibble* (4 bits of data), a *byte* (8 bits of data), or as a *word* (16 bits of data). It should also be noted that, in some instances, a word may be defined as the data width that a given microcontroller can recognize at a time, be it 8 bits or 16 bits. For purposes of this chapter, we will refer to a word as being 16 bits. Data can also be expressed as a double word which is an unsigned 32-bit variable with a value between 0 and 4,294,967,295. Most architectures support this data only for shifts, dividends of a 32-by-16 divide, or for the product of a 16-by-16 multiply.

The most common way of referring to a microcontroller is by the width of its CPU. This indicates the width of data that the CPU can process at a time. A microcontroller with a CPU that can process 8 bits of data at a time is referred to as an 8-bit microcontroller. A microcontroller with a CPU that can process 16 bits of data at a time is referred to as a 16-bit microcontroller. With this in mind, it is easy to see why 16-bit microcontrollers offer higher performance than their 8-bit counterparts. Figure 11.3 illustrates a typical 16-bit CPU dia-

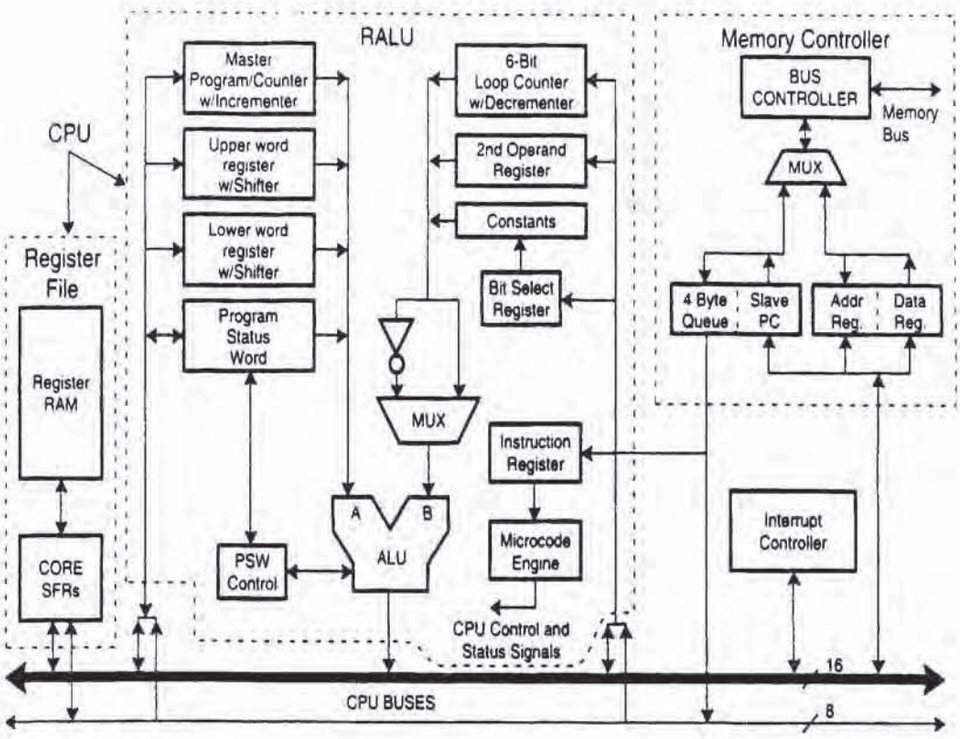


FIGURE 11.3 16-bit CPU.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

gram. The microcode engine controls the CPU. Instructions to the CPU are taken from the instruction queue and temporarily stored in the instruction register. This queue is often referred to as a *prefetch queue* and it decreases execution time by staging instructions to be executed. The microcode engine then decodes the instructions and generates the correct sequence of events to have the ALU perform the desired function(s).

Arithmetic Logic Unit. The ALU is the portion of the CPU that performs most mathematical and logic operations. After an instruction is decoded by the microcode engine, the data specified by the instruction is loaded into the ALU for processing. The ALU then processes the data as specified by the instruction.

Register File. The register file consists of memory locations that are used as temporary storage locations while the user's code is executing. The register file is implemented as RAM and consists of both RAM memory locations and special function registers (SFRs). RAM memory locations are used as temporary data storage during execution of the user's code. After power-up, RAM memory locations default to a logic "0" and data in SFR locations contain default values as specified by the microcontroller manufacturer.

Special Function Registers. SFRs allow the user to configure and monitor various peripherals and functions of the microcontroller. By writing specific data to an SFR, the users can configure the microcontroller to meet the exact needs of their application. Figure 11.4 shows an example of a serial port SFR used for configuration. Note that each bit location within the SFR determines a specific function and can be programmed to either a logic "1" or "0". If more than two configuration choices are possible, two or more bits will be combined to produce the multiple choices. An example of this would be the mode bits (M1 and M2) in the example SFR (Fig. 11.4). Bit locations marked "RSV" are reserved and should be written to with a value as indicated by the manufacturer.

SP_CON (1FBBH)

7	6	5	4	3	2	1	0
0	0	PAR	TB8	REN	PEN	M2	M1

M2, M1	Mode	Function
	00	Mode 0: Synchronous
	01	Mode 1: Standard asynchronous
	10	Mode 2: Asynchronous (receiver interrupt on 9th bit = 1)*
	11	Mode 3: Asynchronous (9th bit = parity or data)**
PEN	Parity Enable. Enables the Parity function for Mode 1 or Mode 3; cannot be enabled for Mode 2.	
REN	Receiver Enable. Enables the receiver to write to SBUF_RX.	
TB8	Transmission Bit 8. Set the ninth data bit for transmission (Modes 2 and 3). Cleared after each transmission; not valid if parity is enabled.	
PAR***	0 = even parity 1 = odd parity	
Bits 6, 7	Reserved; write as zeros for future product compatibility.	

* Mode 2: Asynchronous (receiver: interrupt on 9th bit = 1; transmitter: 9th bit = TB8)
** Mode 3: Asynchronous (receiver: always interrupt on 9th bit; transmitter: 9th bit = parity for PEN = 1
*** Par bit only available on 8XC196KT and KS devices. 9th bit = TB8 for PEN = 0
For 8XC196KR, JR, KQ, JQ devices, this bit should be written as a zero to maintain compatibility with future devices.

FIGURE 11.4 Special function control register example.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

SP_STAT (1FB9H)

7	6	5	4	3	2	1	0
RB8/RPE	RI	TI	FE	TXE	OE	X	X

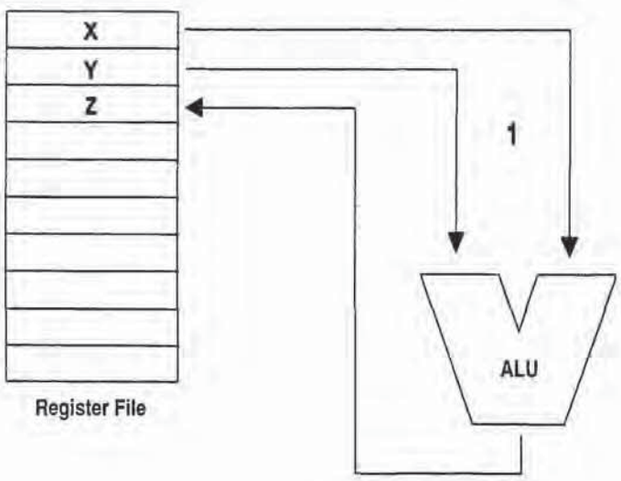
Bits 0, 1	Reserved; ignore data.
OE	Set on buffer overrun error.
TXE	Set on transmitter empty. When set may write 2 bytes to transmit buffer.
FE	Framing error; set it no STOP bit is found at the end of a reception. When set may write 1 byte to transmit buffer.
TI	Transmit interrupt; set at the beginning of the STOP bit transmission.
RI	Receive interrupt; set after the last data bit is received.
RPE	(Parity enabled) Receive parity error (Modes 1 and 3 only); set if parity is enabled and a parity error occurred.
RB8	(Parity disabled) Received Bit 8 (Modes 2 and 3 only); set if the 9th bit is high on reception.

FIGURE 11.5 Special function status register example.

Some SFRs can be read by the user to determine the current status of a given peripheral. Figure 11.5 shows an example of a serial port status register that, when read, indicates the current status of the microcontroller's serial port. Note that each bit location corresponds to a particular state of the serial port. Bit locations marked "RSV" are reserved and should be ignored when read.

Register Direct vs. Accumulator-Based Architectures. Microcontroller architectures can be classified as either the register-direct or accumulator-based type. These terms refer to the means by which the CPU must handle data when performing mathematical, logical, or storage operations.

Register-direct architectures allow the programmer to essentially use most, if not all, of the microcontroller's entire RAM array as individual accumulators. That is, the programmer can perform mathematical or storage operations directly upon any of the RAM locations. This simplifies task switching because program variables may be left in their assigned registers while servicing interrupts. Figure 11.6 illustrates a register-to-register type architecture (such



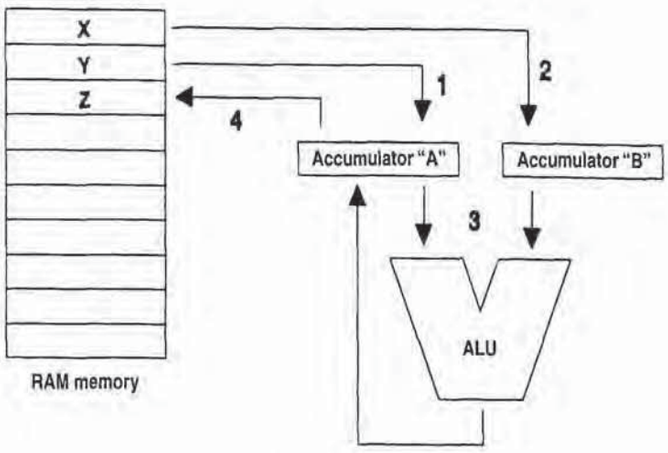

```
Z=X+Y
1. MUL Z,X,Y
```

FIGURE 11.6 Register-to-register architecture example.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

as Intel's MCS®-96). This architecture essentially has 232 "accumulators" (more are available through a windowing mechanism) of which any can be operated on directly by the RALU. The true advantage of this type of architecture is that it reduces accumulator bottleneck and speeds throughput during program execution.

Accumulator-based architectures require the user to first store the data to be manipulated into a temporary storage location, referred to as an "accumulator," prior to performing any type of data operation. After the operation is completed, the user program must then store the result to the desired destination location. Figure 11.7 depicts an example of an accumulator-based architecture.




```
Z=X+Y
1. LD A,Y
2. LD B,X
3. MUL A,B
4. ST A,Z
```

FIGURE 11.7 Accumulator-based architecture.

Program Counter. The Program Counter (PC) controls the sequencing of instructions to be executed. The PC is a 16-bit register located within the CPU which holds the address of the next instruction to be executed. After an instruction is fetched, the PC is automatically incremented to point to the next instruction.

SP starting address (SP+12):	
(SP+10):	80FFh
(SP+8):	A5A5h
(SP+6):	6E20h
(SP+4):	5555h
(SP+2):	0000h
SP ending address (SP):	8000h

FIGURE 11.8 Stack pointer example.

Stack and Stack Pointer. The stack is an area of memory (typically user-assigned) that is used to store data temporarily in a FILO (first-in, last-out) fashion. The stack is primarily used for storing program information (such as the program counter or interrupt mask registers) when an interrupt service routine is invoked. It is also sometimes used to pass variables between subroutines. The stack is typically accessed through PUSH and POP instructions. Execution of the PUSH instruction "pushes" the contents of the specified operand onto the stack whereas the

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

POP instruction "pops" the contents of the specified operand off of the stack. The stack pointer (SP) is a register which points to the next available word location on the stack. Consider the example shown in Fig. 11.8 which shows the contents of the stack after the following code sequence is executed:

PUSH #80FFh	pushes immediate data 80FFh onto stack
PUSH #0A5A5h	pushes immediate data A5A5h onto stack
PUSH 82h	pushes data @ 82h (assume it's 6E20h) onto stack
PUSH #5555h	pushes immediate data 5555h onto stack
PUSH 4Eh	pushes data @ 4Eh (assume it's 0000h) onto stack
PUSH #8000h	pushes immediate data 8000h onto stack

Continuing with the preceding example, if a POP instruction were executed, the data at the current SP address (SP) would be "popped" off the stack and stored to the address specified by the instruction's operand. Executing the POP instruction results in the SP being incremented by 2.

Program Status Word and Flags. The program status word (PSW) is a collection of boolean flags which retain information concerning the state of the user's program. These flags are set or cleared depending upon the result obtained after executing certain instructions as specified by the microcontroller manufacturer. PSW flags are not directly accessible by the user's program; access is typically through instructions which test one or more of the flags to determine proper program flow. Following is a summary of common PSW flags as supported by Intel's MCS-96® architecture:

Z: The *Zero* flag is set when an operation generates a result equal to zero. The Z flag is never set by the add-with-carry (ADDC/ADDCB) or subtract-with-carry (SUBC/SUBCB) instructions, but is cleared if the result is nonzero. These two instructions are normally used in conjunction with ADD/ADDB and SUB/SUBB instructions to perform multiple-precision arithmetic. The operation of the Z flag for these instructions leaves it indicating the proper result for the entire multiple-precision calculation.

N: The *Negative* flag is set when an operation generates a negative result. Note that the N flag will be in the algebraically correct state even if overflow occurs. For shift operations, the N flag is set to the same value as the most significant bit of the result.

V: The *oVerflow* flag is set when an operation generates a result that is outside the range for the destination data type. For shift-left instructions, the V flag is set if the most significant bit of the operand changes at any time during the shift. For an unsigned word divide, the V flag is set if the quotient is greater than 65,535. For a signed word divide, the V flag is set if the quotient is less than –32,768 or greater than 32,767.

VT: The *oVerflow Trap* flag is set when the V flag is set, but it is only cleared by instructions which are specially designated to clear the VT flag (such as CLRVT, JVT, and JNVT). The VT flag allows for testing possible overflow conditions at the end of a sequence of related arithmetic operations. This is normally more efficient than testing the V flag after each instruction.

C: The *Carry* flag is set to indicate either (1) the state of the arithmetic carry from the most significant bit of the ALU for an arithmetic operation or (2) the state of the last bit shifted out of an operand for a shift. Arithmetic borrow after a subtract operation is the complement of the C flag (i.e., if the operation generated a borrow, then C = 0).

ST: The *STicky* bit flag is set to indicate that, during a right shift, a 1 has been shifted first into the C flag and then shifted out. The ST flag can be used along with the C flag to control rounding after a right shift. Imprecise rounding can be a major source of error in a numerical calculation; use of both the C and ST flags can increase accuracy as described in the following paragraphs.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Consider multiplying two 8-bit quantities and then scaling the result down to 12 bits:

```
MULUB   AX, CL, DL    (CL * DL = AX)
SHR     AX, #4        (AX is shifted right by 4 bits)
```

If the C flag is set after the shift, it indicates that the bits shifted off the end of the operand were greater than or equal to one-half the least significant bit of the 12-bit result. If the C flag is cleared after the shift, it indicates that the bits shifted off the end of the operand were less than half the LSB of the 12-bit result. Without the ST flag, the rounding decision must be made on the basis of the C flag alone. (Normally the result would be rounded up if the C flag is set.) The ST flag allows a finer resolution in the rounding decision as shown here:

C	ST	Bits shifted off
0	0	Value = 0
0	1	0 < Value < ½ LSB
1	0	Value = ½ LSB
1	1	Value > ½ LSB

Jump instructions are the most common instructions to utilize PSW flags for determining the operation to perform. Instructions that test PSW flags are very useful when program flow needs to be altered dependent upon the outcome of an arithmetic operation. The most common example of this would be for program loops that are to be executed a certain number of times. Following are examples of several MCS-96 instructions whose operation is dependent upon the state of one or more program status word flags:

JC (Jump if C flag is set.) If the C (carry) bit is set, the program will jump to the address location specified by the operand. If the C flag is cleared, control will pass to the next sequential instruction.

JGT (Jump if signed greater than.) If both the N (negative) and the Z (zero) flags are clear, the program will jump to the address location specified by the operand. If either of the flags is set, control will pass to the next sequential instruction.

JLE (Jump if signed less than or equal.) If either the N or Z flags is set, the program will jump to the address location specified by the operand. If both the N and Z flags are cleared, control will pass to the next sequential instruction.

11.1.4 Bus Controller

The bus controller serves as the interface between the CPU and the internal program memory and the external memory spaces. The bus controller maintains a queue (commonly called the prefetch queue) of prefetched instruction bytes and responds to CPU requests for data memory references. The prefetch queue decreases execution time by staging instructions to be executed. The capacities of prefetch queues vary but for the MCS-96 architecture, it is 4 bytes deep.

When using a logic analyzer to debug code it is important to consider the effects of the prefetch queue. It is not possible to accurately determine when an instruction will execute by simply watching when it is fetched from external memory. This is because the prefetch queue is filled in advance of instruction execution. It is also important to consider the effects when a jump or branch occurs during program execution. When the program sequence changes because of a jump, interrupt, call, or return, the PC is loaded with the new address, the queue is flushed and processing continues. Consider the situation in which the external address/data bus is being monitored when a program branch occurs. Because of the prefetch queue, it will appear as if instructions past the branch point were executed, when in fact they were only loaded into the prefetch queue.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

11.1.5 Frequency of Operation

Microcontrollers are being offered in an ever-increasing range of operating frequencies. Most high-end automotive applications currently use microcontrollers operating in the 12- to 20-MHz range, with 24 MHz becoming not so uncommon. Microcontrollers with frequencies as high as 30 and 32 MHz are available as prototypes and will soon be available for production. Operating frequency becomes especially important when a microcontroller must perform high-speed event control such as required in ABS braking and engine control. Applications such as these typically have to detect, calculate, and respond to external events within a given amount of time. In ABS applications, this time is commonly referred to as loop time and defines the amount of time that the microcontroller has to execute the main loop of the software algorithm to achieve optimal performance.

Operating frequency can be directly related to the speed at which a microcontroller will execute the user's code. For instance, let's look at how long it takes for a particular microcontroller to execute the following generic subroutine. For this example, consider the execution times rather than the operations each instruction is performing.

```
(6)  PUSHF
(3)  NOTB   PTS_COUNT_EPA1
(5)  ADDB   NUM_OF_PULSES_1, PTS_COUNT_EPA1, #00h
(5)  SUB    INV_SPEED_1, FTIME_1, ITIME_1
(27) DIV    INV_SPEED_1, NUM_OF_PULSES_1
(5)  LD     Temp1+2, #Speed_high_constant
(5)  LD     Temp1, #Speed_low_constant
(27) DIV    Temp1, INV_SPEED_1
(4)  ST     Temp1, EPA1_FREQ
(11) RET
```

In this example, the numbers in brackets {} denote how many state times it will take the microcontroller to execute the given line of code. A state time is the basic time measurement for all microcontroller operations. For this MCS-96 family microcontroller, a state time is based on the crystal frequency divided by two. A state time for other microcontrollers may be based upon the crystal frequency divided by three. For this particular microcontroller, a state time can be calculated by the following formula (other microcontroller families use similar formulas):

$$1 \text{ state time} = 1[(\text{frequency of operation})/2]$$

Applying this formula, 1 state time = 125 ns when operating at 16 MHz, and 167 ns when operating at 12 MHz. The example code sequence takes the microcontroller 98 state times to execute. This equates to 16.37 μs to execute at an operating frequency of 12 MHz. At 16 MHz, it takes only 12.25 μs for the microcontroller to execute the subroutine. An operating frequency of 16 MHz results in the microcontroller executing approximately 34 percent more instructions in a given time than at a frequency of 12 MHz.

Another consideration when choosing an operating frequency is the clocking resolution of on-chip timer/counters. The maximum clocking rate of on-chip timer/counters is limited by the frequency the microcontroller is being clocked at. As an example, if an on-chip timer/counter is set up to increment/decrement at a rate determined by CLOCK/4, this would result in 333 ns resolution at 12 MHz. However, if the clock speed were increased to 16 MHz, a higher and more desirable resolution of 250 ns is achieved.

11.1.6 Instruction Set

An often overlooked feature that gives a microcontroller the capability to perform desired operations and manipulate data is its instruction set. A microcontroller's instruction set con-

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

sists of a set of unique commands which the programmer uses to instruct the microcontroller on what operation to perform.

An *instruction* is a binary command which is recognized by the CPU as a request to perform a certain operation. Examples of typically supported operations are loads, moves, and stores which transfer data from one memory location to another. There are also jumps and branches which are used to alter program flow. Arithmetic instructions include various multiples, divides, subtracts, additions, increments, and decrements. Instructions such as ANDs, ORs, XORs, shifts, and so forth, allow the user to perform logical operations upon data. In addition to these basic instructions, microcontrollers often support specialized instructions unique to their architecture or intended application.

Instructions can be divided into two parts, the *opcode* and *operand.* The opcode (sometimes referred to as the machine instruction) specifies the operation to take place and the operand specifies the data to be operated upon. Instructions typically consist of either 0, 1, 2, or 3 operands to support various operations. As an example, consider the following MCS-96 architecture instructions:

PUSHF (0 operands) is an instruction that pushes the program status word (PSW) onto the stack. Since this instruction operates on a predefined location, no operand is necessary.

Format: PUSHF

PUSH (1 operand) is an instruction that pushes the specified word operand onto the stack.

Format: PUSH (SRC)

ADD (2 operands) adds two words together and places the result in the destination (leftmost) operand location.

Format: ADD (DST),(SRC)

ADD (3 operands) adds two words together as the 2-operand ADD instruction, but in this case, a third operand is specified as the destination.

Format: ADD (DEST),(SRC1),(SRC2)

Instructions support one or more of six basic addressing types to access operands within the address space of the microcontroller. If programmers wish to take full advantage of a microcontroller architecture, it is important that they fully understand the details of the supported addressing types. The six basic types of addressing modes are termed register-direct, indirect, indirect with autoincrement, immediate, short-indexed, and long-indexed. The following descriptions describe these modes as they are handled by hardware in register-to-register architectures.

The *register-direct* addressing mode is used to directly access registers within the lower 256 bytes of the on-chip register file. The register is selected by an 8-bit field within the instruction and the register address must conform to the operand type's alignment rules. Depending upon the instruction, typically up to three registers can take part in the calculation.

Examples:

ADD AX,BX,CX	AX = BX + CX
MUL AX,BX	AX = AX*BX
INCB CL	CL = CL + 1

The *indirect addressing* mode accesses a word in the lower register file containing the 16-bit operand address. The indirect address can refer to an operand anywhere within the address space of the microcontroller. The register containing the indirect address is selected by an 8-bit field within the instruction. An instruction may contain only one indirect reference; the remaining operands (if any) must be register-direct references.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Examples:

LD BX,[AX] BX = mem_word(AX)

In this example, assume that before execution:

contents of AX = 2FC2h

contents of 2FC2h = 3F26h

Then after execution,

contents of BX = 3F26h

ADDB AL,BL,[CX] AL = BL + mem_byte(CX)

The *indirect with autoincrement* addressing mode is the same as the indirect mode except that the variable that contains the indirect address is autoincremented after it is used to address the operand. If the instruction operates on bytes or short integers, the indirect address variable is incremented by one; if it operates on words or integers, the indirect address will be incremented by two.

Examples:

LD BX,[AX]+ BX = mem_word(AX)
AX = AX + 2

ADDB AL,BL,[CX]+ AL = BL + mem_byte(CX)
CX = CX + 1

For the *immediate addressing* mode, an operand itself is in a field in the instruction. An instruction may contain only one immediate reference; the remaining operand(s) must be register-direct references.

Example:

ADD AX,#340 AX = AX + 340 (decimal)

For the *short-indexed addressing* mode, an 8-bit field in the instruction selects a word variable (which is contained in square brackets) in the lower register file that contains an address. A second 8-bit field in the instruction stream is sign-extended and summed with the word variable to form an operand address.

Since the 8-bit field is sign-extended, the effective address can be up to 128 bytes before the address in the word variable and up to 127 bytes after it. An instruction may contain only one short-indexed reference; the remaining operand(s) must be register-direct references.

Example:

LD AX,4[BX] AX = mem_word(BX + 4)

In this example, assume that before execution:

contents of BX = A152h

The operand address is then A152h + 04h = A156h

The *long-indexed addressing* mode is like the short-indexed mode except that a 16-bit field is taken from the instruction and added to the word variable to form the operand. No sign extension is necessary. An instruction may contain only one long-indexed reference and the remaining operand(s) must be register-direct references.

Examples:

ST AX,TABLE[BX] mem_word(TABLE + BX) = AX

AND AX,BX,TABLE[CX] AX = BX and mem_word(TABLE + CX)

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

11.1.7 Programming Languages

The two most common types of programming languages in use today for automotive microcontrollers are *assembly languages* and *high-level languages* (HLLs). Program development begins with the user writing code in either an assembly language or an HLL. This code is written as a text file and is referred to as a source file. The source file is then assembled or compiled using the appropriate assembler/compiler program. The assembler translates the source code into object code and creates what is referred to as an object file. The object file contains machine language instructions and data that can be loaded into an evaluation tool for debugging and validation. The object can also be converted into a hex file for EPROM programming or ROM mask generation as discussed later in this chapter. The program development process is illustrated in Fig. 11.9.

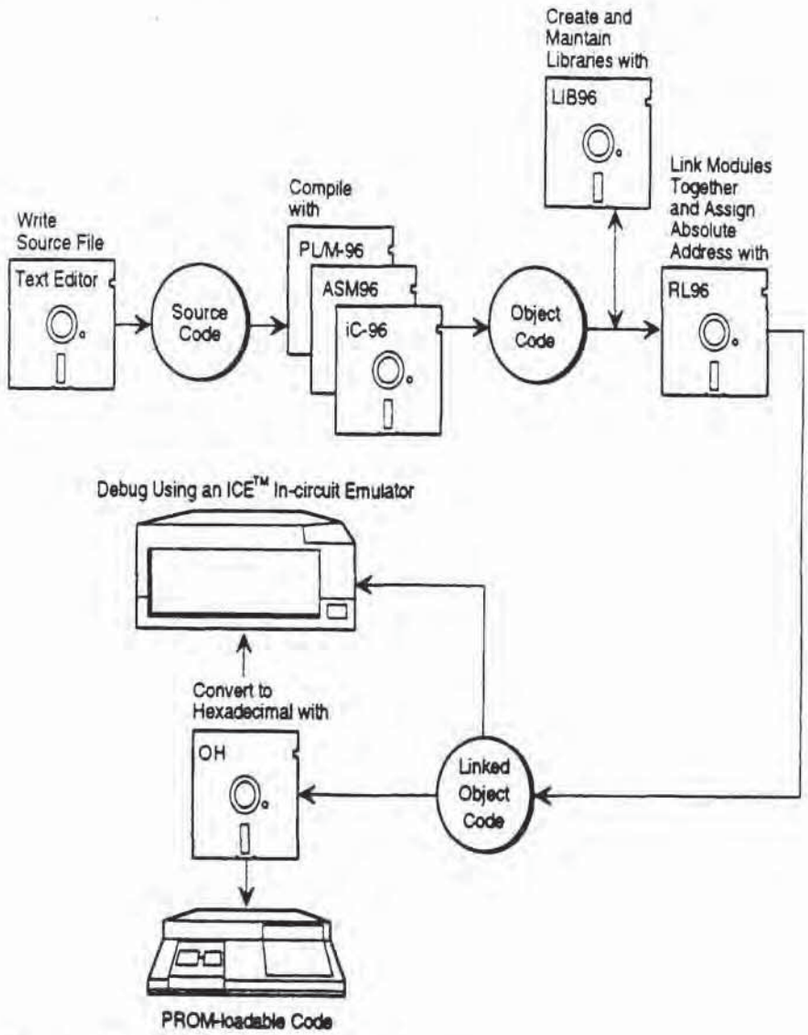


FIGURE 11.9 The program development process.

Assembly Language Programming. An assembly language is a low-level programming language that is specific to a given microcontroller family. Assemblers translate language operation codes (mnemonics) directly into machine instructions that instruct the microcontroller

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

on what operation to perform. Because the programmer is essentially using the microcontroller's machine code to write assembly language programs, more precise control of the device can be achieved through the direct manipulation of individual bits within registers. Because of their efficiency, assembly language programs require less code space than high-level languages. Assembly language programs consist of three parts: machine instructions, assembler directives, and assembler controls.

A *machine instruction* is a machine code that can be executed by the microcontroller's CPU. The collection of machine instructions that a particular microcontroller can execute is referred to as its instruction set. An example of a machine instruction is the opcode for the MULB instruction (Fig. 11.10) from Intel's MCS-96 assembly language. MULB is the mnemonic that represents the machine instruction which performs the specified multiplication operation. When executed by the microcontroller, the MULB opcode results in the multiplication of the two byte operands with the result being placed in a word destination location.

MULB (Three Operands)

Format

```
MULB   wreg,breg,baop
```

Operation

The second and third byte operands are multiplied using signed arithmetic and the 16-bit result is stored into the destination (leftmost) operand. The sticky bit flag is undefined after the instruction is executed.

```
(DEST) ← (SRC1) * (SRC2)
```

Opcode Pattern

11111110	01011aa	baop	breg	wreg

Flags Affected ST

Examples

```
MULB    DELTA, TIMER1, #2
MULB    ALPHA, BETA, GAMMA
MULB    ALPHA, DELTA, 10[GAMMA]
```

FIGURE 11.10 Machine instruction example: MULB.

Assembler directives allow the user to specify auxiliary information (such as storage reservation, location counter control, definition of nonexecutable code, object code relocation, and flow of assembler processing) that determines the manner in which the assembler generates object code from the user's source file input.

Assembler controls set the mode of operation for the assembler and direct the flow of the assembly process. Assembler controls can be classified into primary controls and general controls. Primary controls are set at the beginning of the assembly process and cannot be changed during the assembly. Primary controls allow the user to specify items such as print options, page lengths and widths, error messages, and cross-referencing. General controls can be specified in the invocation line or on control lines anywhere in the source file and can appear any number of times in the program. General controls either cause an immediate action or an immediate change of conditions in which the condition specified remains in effect until another general control causes it to change.

High-level Language Programming. Unlike low-level languages (such as assembly languages), a high-level language is a general purpose language that can support numerous microcontroller architectures. The most common high-level language used for automotive

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

applications is C. C programs are written with statements rather than specific instructions from a microcontroller's instruction set. High-level languages utilize a software program known as a *compiler* to translate the user's source code into the specific microcontroller's machine language. Each microcontroller family has its own unique compiler to support selected high-level languages. Although high-level languages tend to be less efficient than assembly languages, their advantage lies in ease of writing code and better debugging capability. The use of statements as opposed to specific instructions better suits high-level languages toward control of procedures (to implement complex software algorithms) as opposed to the microcontroller itself.

11.1.8 Interrupt Structure

The interrupt structure is one of the more important features of an automotive microcontroller. Applications such as automotive ABS and engine control can be referred to as event-driven control systems. Event-driven control systems require that normal code execution be halted to allow a higher-priority task or event to take place. These higher-priority tasks are known as interrupts and can initiate a change in the program flow to execute a specialized routine. When an interrupt occurs, instead of executing the next instruction, the CPU branches to an interrupt service routine (ISR). The branch can occur in response to a request from an on-chip peripheral, an external signal, or an instruction. In the simplest case, the microcontroller receives the request, performs the desired operation and returns to the task that was interrupted.

ISRs are typically serviced via software but it is becoming common for microcontroller manufacturers to implement special on-chip hardware ISR functions for commonly performed operations. These ISRs are typically microcoded or *hardwired* into the microcontroller as described later in this section.

Software, or Normal, Servicing of Interrupts. The software servicing of interrupts is fairly straightforward as shown in Fig. 11.11. When an interrupt source is enabled by the user and a

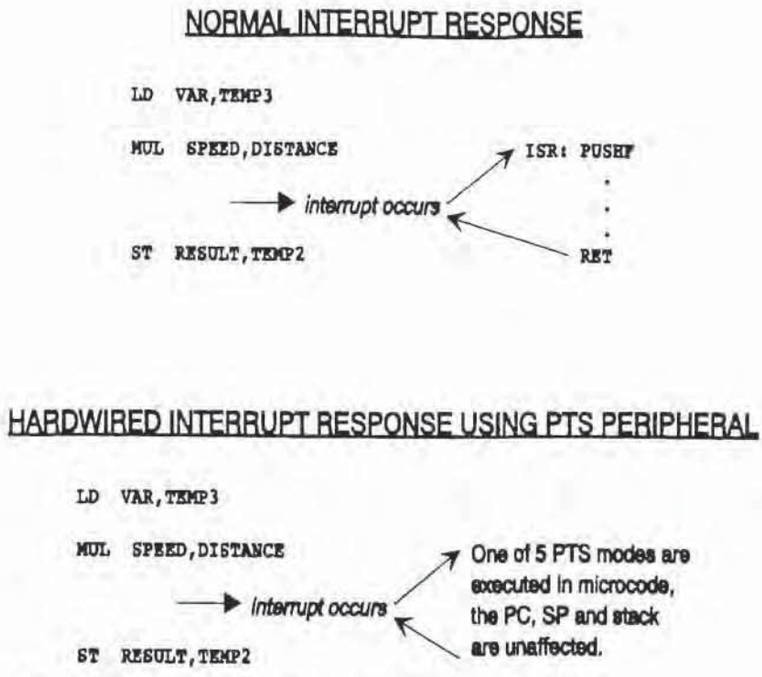


FIGURE 11.11 Comparison of normal interrupts and hardwired interrupts.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

valid interrupt event occurs, the CPU will fetch the starting address of the ISR from the interrupt vector table. The interrupt vector table is a dedicated section of memory that contains the user-programmed start address of the various ISRs. After fetching the ISR address, the CPU automatically pushes the current program counter (PC) onto the stack and loads the PC with the ISR beginning address. This results in the program flow vectoring to the ISR address. The user-programmed ISR is then executed. The last instruction within the ISR is a return instruction that pops the old PC off the stack. This results in program flow continuing from where it was interrupted.

Interrupt mask registers allow the user to prevent or *mask* undesirable interrupts from occurring during various sections of the program. This is a very desirable feature and allows for custom tailoring of the interrupt structure to meet the needs of a particular application. Enabling or disabling of all interrupts (known as globally enabling/disabling) is typically supported with a software instruction such as DI (globally disable all interrupts) or EI (globally enable all interrupts).

Hardware, or Microcoded, Interrupt Structures. Hardware interrupt structures differ from software interrupts in that the user doesn't have to provide the ISR to be executed when the interrupt occurs. With a hardware interrupt structure, the ISR is predefined by being hardwired or *microcoded* into the microcontroller. This is advantageous because it requires less code space and requires less CPU overhead. Stack operations are not necessary since interrupt vectors do not have to be fetched. Most microcontroller manufacturers have their own proprietary solution for hardware ISR's, which are all somewhat similar to one another. For purposes of this section, we will briefly describe the peripheral transaction server as implemented on members of Intel's MCS-96 family of microcontrollers.

The PTS provides a microcoded hardware interrupt handler which can be used in place of a normal ISR. The PTS requires much less overhead than a normal ISR since it operates without modification of the stack. Any interrupt source can be selected by the user to trigger a PTS interrupt in place of a normal ISR. The PTS is similar to a direct memory access (DMA) controller in that when a PTS interrupt, or *cycle*, occurs, data is automatically moved from one location of memory to another as specified by the user. Figure 11.11 compares a regular ISR to a PTS interrupt cycle.

The PTS allows for five modes of operation; single-byte transfer, multiple-byte transfer, PWM, PWM toggle, and A/D scan mode. Each mode is configurable through an 8-byte, user-defined PTS control block (PTSCB) located in RAM. The user may enable virtually any normal interrupt source to be serviced by a PTS interrupt by simply writing to the appropriate bit in an SFR known as the PTS_SELECT register. When a PTS interrupt is enabled and the event occurs, a microcoded interrupt service routine executes in which the contents of the PTSCB are read to determine the specific operation to be performed. More details on the PTSCB can be found in the application example found in this section.

The major advantage of the PTS for automotive applications is its fast response time. The PTS is ideally suited for transferring single or multiple bytes/words of data in response to an interrupt. An example of this is the serial port example which will be described shortly. Another example of the usefulness of the PTS (using A/D scan mode) would be if the user wanted to automatically store A/D conversion results every time a conversion completed within a user-defined scan of A/D channels. The PTS could also be configured to automatically transfer a block of data between memory locations every time an interrupt occurs.

Application Example of PTS Single-Byte Transfer Mode. This example shows how the PTS can be used to automatically transmit and receive 8-byte messages over the serial port. Data to be transmitted and received data are stored in separate tables. The use of the PTS for this purpose greatly reduces CPU overhead and code-space requirements. The layout of the user-defined PTSCB for single-byte transfer mode is shown in Fig. 11.12. PTS_DEST within the PTSCB contains the destination address for the data transfer and PTS_SOURCE contains the source address for the transfer.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

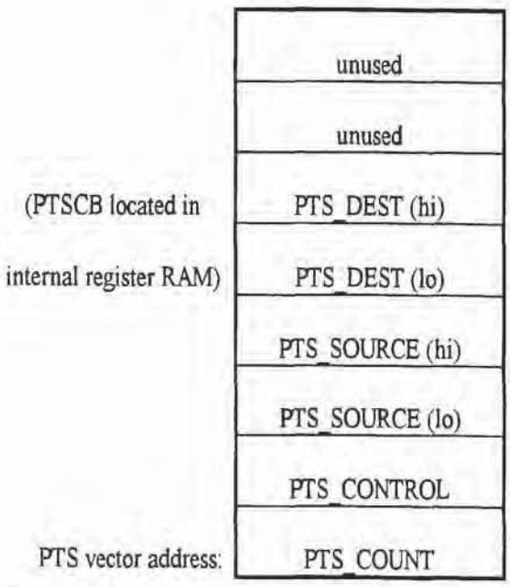


FIGURE 11.12 PTS control block for single-byte transfer mode.

Two PTSCBs are set up for this example, one in response to receive (RX) interrupts and one in response to transmit (TX) interrupts. The RX PTSCB's PTS_DEST is initialized with the start address of the receive data table and the TX PTSCB's PTS_DEST is initialized with the address of the serial port's transmit buffer.

PTS_CONTROL is a byte that specifies the PTS operation to be performed. Its layout is shown in Fig. 11.13.

PTS_COUNT is a down counter that is used to keep track of how many PTS interrupts or cycles have occurred since the last initialization. PTS_COUNT is initialized by the user to any value below 256 and is decremented everytime the corresponding PTS cycle occurs. It is often used to keep track of how many pieces of data have been transferred. In this example, PTS_COUNT is used to determine when a complete 8-byte message has been transmitted or received. After PTS_COUNT expires, an "end-of-PTS" or "normal" ISR occurs, in which the user utilizes the data as required by the application. When an interrupt source is enabled by the user to be a PTS interrupt, the following sequence of events occurs every time the corresponding interrupt occurs:

1. Instead of a normal interrupt, the user has selected it to do a PTS cycle.
2. The microcoded PTS routine fetches the PTS_CONTROL byte from the PTSCB whose start address is specified by the user in the PTS interrupt vector table. The microcoded PTS routine then:

   reads data to be transferred from address specified by PTS_SOURCE

   writes the data to address specified by PTS_DEST

   optionally increments/updates PTS_SOURCE and PTS_DEST addresses

   decrements PTS_COUNT
3. When PTS_COUNT reaches "0", an end of PTS interrupt occurs and the normal ISR is executed in which the user utilizes the received data as necessary (for RX interrupts) or reloads the transmit table with new data (for TX interrupts).

Interrupt Latency. Interrupt latency is defined as the time from when the interrupt event occurs (not when it is acknowledged) to when the microcontroller begins executing the first

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

PTS_CONTROL BYTE (for single and multiple-byte transfers)

7	6	5	4	3	2	1	0
M2	M1	M0	B/W	SU	DU	SI	DI

M2, M1, M0:	Mode	Function
	000	PTS Block Transfer
	100	PTS Single Transfer

B/W:	Byte/Word:	"0" = Word; "1" = Byte
SU:	Source Update:	"1" = update source address
DU:	Destination Update:	"1" = update destination address
SI:	Increment Source:	"1" = Increment Source address
DI:	Increment Destination:	"1" = Increment Destination address

FIGURE 11.13 PTS control byte for single- and multiple-byte transfer modes.

instruction of the interrupt service routine. Interrupt latency must be carefully considered in timing-critical code as is found in many automotive applications.

There is a delay between an interrupt's triggering and its acknowledgment. An interrupt is not acknowledged until the currently executing instruction is finished. Further, if the interrupt signal does not occur at least some specified (assume four for this discussion) state times before the end of the current instruction, the interrupt may not be acknowledged until after the next instruction has been executed. This is because an instruction is fetched and prepared for execution a few state times before it is actually executed. Thus, the maximum delay between interrupt generation and its acknowledgment is approximately four state times plus the execution time of the next instruction.

It should also be noted that most microcontrollers have protected instructions (such as RETURN, PUSH, POP) which inhibit interrupt acknowledgment until after the following instruction is executed. These instructions can increase interrupt-to-acknowledgment delay.

When an interrupt is acknowledged, the interrupt pending bit is cleared and a call is forced to the location indicated by the corresponding interrupt vector. This call occurs after the completion of the current instruction, except as noted previously. For the MCS-96 architecture, the procedure of fetching the interrupt vector and forcing the call requires 16 state times. The stack being located in external memory will add an additional two state times to this number.

Latency is the time from when an interrupt is generated (not acknowledged) until the microcontroller begins executing interrupt code. The maximum latency occurs when an inter-

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

rupt occurs too late for acknowledgment following the current instruction. The worst case is calculated assuming that the current instruction is not a protected one. The worst-case latency is the sum of three terms:

1. The time for the current instruction to finish (assume four state times).
2. The state times required for the next instruction. This time is basically the time it takes to execute the longest instruction used in the user's code (assume it's a 16-state DIV instruction).
3. The response time (assume 16 states, 18 for an externally located stack).

Thus, for this scenario, the maximum delay would be 4 + 16 + 16 = 36 state times. This equates to approximately 4.5 μs for a MCS-96 microcontroller operating at 16 MHz. This latency can increase or decrease depending upon the longest execution-time instruction used. Figure 11.14 illustrates an example of this worst-case scenario.

Interrupt latency can be reduced by carefully selecting instructions in areas of code where interrupts are expected. Using a protected instruction followed immediately by a long instruction increases the maximum latency because an interrupt cannot occur after the protected instruction.

11.1.9 Fabrication Processes

The basic fabrication processes that are widely used for automotive microcontrollers today are NMOS (N-channel metal-oxide semiconductor) and CMOS (complementary MOS). The scope of this chapter does not allow for an in-depth discussion of these processes, although a brief description of the structures used to build on-chip circuitry will be discussed. These terms refer to the components used in the construction of MOSFET (MOS field effect transistor) inverters which are the basis of logic on digital devices. NMOS inverters are constructed of N-channel transistors only, whereas CMOS inverters are constructed of both N-channel and P-channel transistors. This section will describe the basic operation of each inverter along with its pros and cons.

Simply stated, a P-channel transistor conducts when a logic "0" is applied to its gate. Conversely, N-channel transistors conduct when a logic "1" is applied to their gate. Figures 11.15 and 11.16 show a simplified cross-sectional view and the electrical symbol for N- and P-channel devices, respectively.

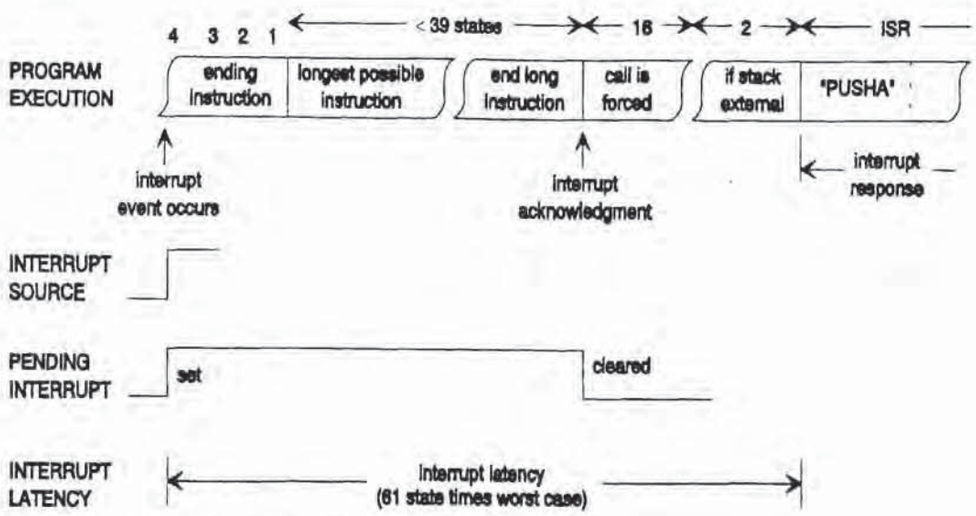


FIGURE 11.14 Worst case interrupt latency example.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

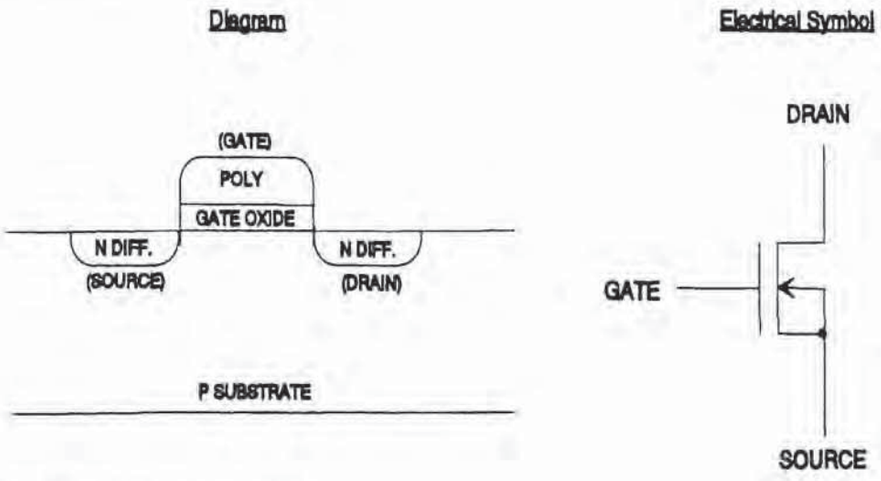


FIGURE 11.15 N-channel transistor.

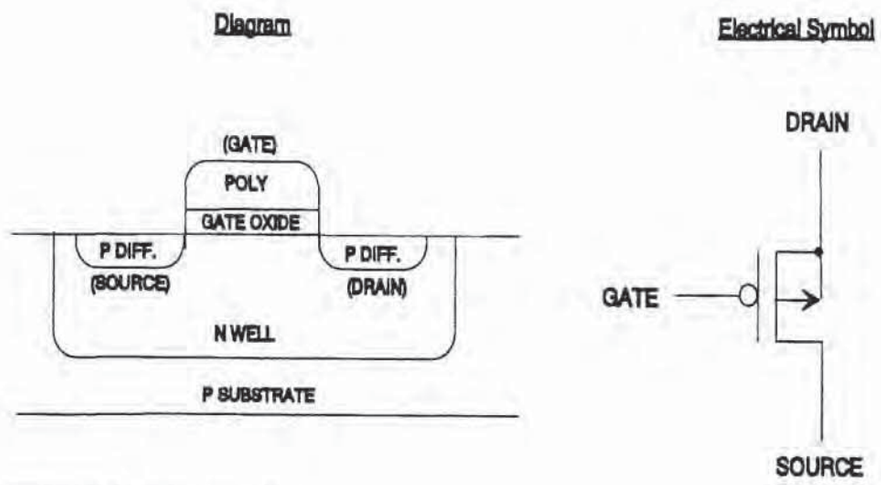


FIGURE 11.16 P-channel transistor.

NMOS Inverters. NMOS inverters are constructed of two NMOS transistors in which one is utilized as a resistance (Q2) and the other is utilized as a switch (Q1). A depletion-mode NMOS transistor is commonly utilized for the resistance device. A basic NMOS inverter is shown in Fig. 11.17. Note that Q2 is always on and acts as a resistor.

When a logic "0" is applied to the inverter's input, Q1 is turned off, which results in Q2 driving a logic 1 at the output. When a logic "1" is applied to the inverter's input, Q1 is turned on and overcomes Q2. This results in a logic "0" at the output.

NMOS microcontrollers are still produced in large quantities today. An advantage of NMOS processes is the simplistic circuit configuration which results in higher chip densities. NMOS devices are also less sensitive to electrostatic discharge (ESD) than CMOS devices. An inherent disadvantage of NMOS design is the slower switching speeds and higher power dissipation due to the dc current path from power to ground through Q1 and Q2 when the inverter is driving a logic "0".

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

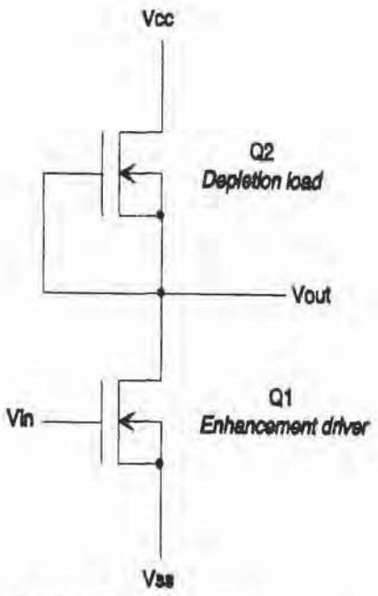


FIGURE 11.17 NMOS inverter.

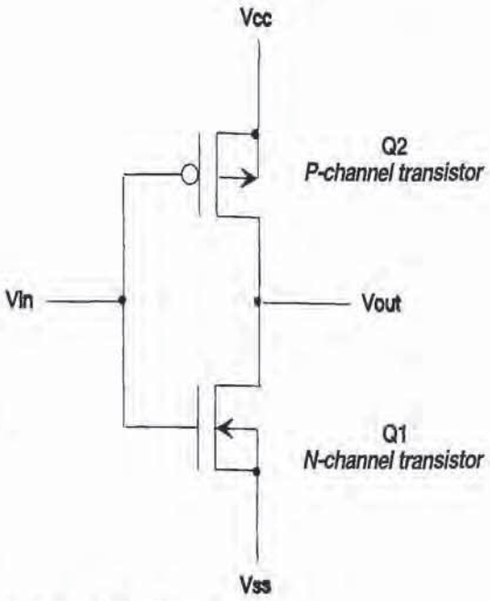


FIGURE 11.18 CMOS inverter.

CMOS Inverters. The CMOS is the most widely used process for automotive microcontrollers today. CMOS inverters are constructed of both P-channel and N-channel transistors that have their inputs tied together as shown in Fig. 11.18. When a logic "0" is applied to the inverter's input, Q1 is turned off and Q2 is turned on, which results in Q2 driving a logic "1" at the output. When a logic "1" is applied to the inverter's input, Q2 is turned off and Q1 is turned on, which results in Q1 driving a logic "0" at the output. Note that only one of these two devices will conduct at a time when the input is "1" or "0". While the input switches, both Q1 and Q2 may conduct for a short time resulting in a small amount of power dissipation.

The main advantages of CMOS logic are greatly improved switching times and lower power consumption, which is due to the complementary design of the inverter. A disadvantage of CMOS logic is that it is more expensive due to its increased complexity and more demanding fabrication process. CMOS logic is more susceptible to ESD damage, although microcontroller manufacturers have countered this by incorporating very effective ESD protection devices onto the silicon.

11.1.10 Temperature Range

Another important factor that must be considered when choosing a microcontroller is the temperature range in which it will be required to operate. The two most common temperature specifications specified by microcontroller manufacturers are *ambient temperature under bias* (TA) and *storage temperature*. These specifications are based upon package thermal characteristics as determined through device and package testing. Storage temperature refers to the temperature range that a microcontroller can be subjected to during periods of nonoperation. Storage temperature specifications are more extreme than ambient temperature under bias temperatures and are usually all the same regardless of the specified ambient temperature range. The common storage temperature range in industry is −60 to +150 °C. While powered-down, a given microcontroller must not be subjected to temperatures that exceed its specified storage temperature range.

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Ambient temperature under bias (TA) refers to the temperature range that the microcontroller is guaranteed to operate at within a given application. While powered-up or operating, a microcontroller must not be subjected to temperatures that exceed its specified ambient temperature range. The most common ambient temperature ranges in industry are:

Commercial	0 to +70 °C
Extended	–40 to +85 °C
Automotive	–40 to +125 °C

11.2 MEMORY

Microcontrollers execute customized programs that are written by the user. These programs are stored in either on-chip or off-chip memory and are often referred to as the *user's code*. On-chip memory is actually integrated onto the same piece of silicon as the microcontroller and is accessed over the internal data bus. Off-chip memory exists on a separately packaged piece of silicon and is typically accessed by the microcontroller over an external address/data bus.

A memory map shows how memory addresses are arranged in a particular microcontroller. Figure 11.19 shows a typical microcontroller memory map.

<table>
<tr><th>Address</th><th colspan="3">Memory Function</th></tr>
<tr><td>0FFFFh
0A000h</td><td colspan="3">External Memory</td></tr>
<tr><td>9FFFh
2080h</td><td colspan="3">Internal ROM/EPROM or External Memory</td></tr>
<tr><td>207Fh
2000h</td><td colspan="3">Internal ROM/EPROM or External Memory
(Interrupt vectors, CCB's, Security Key, Reserved locations, etc.)</td></tr>
<tr><td>1FFFh
1F00h</td><td colspan="3">Internal Special Function Registers (SFR's)</td></tr>
<tr><td>1EFFh
0600h</td><td colspan="3">External Memory</td></tr>
<tr><td>05FFh
0400h</td><td colspan="3">INTERNAL RAM (Address with indirect or indexed modes.)
(Also know as Code RAM)</td></tr>
<tr><td>03FFh
0100h</td><td>Register RAM</td><td>Upper Register File (Address with indirect or indexed modes or through windows.)</td><td rowspan="3">Register File</td></tr>
<tr><td>00FFh
0018h</td><td>Register RAM</td><td rowspan="2">Lower Register File (Address with direct, indirect or indexed modes.)</td></tr>
<tr><td>0017h
0000h</td><td>CPU SFRs</td></tr>
</table>

FIGURE 11.19 Microcontroller memory map.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Memory is commonly referred to in terms of Kbytes of memory. One Kbyte is defined as 1024 bytes of data. Memory is most commonly arranged in bytes which consist of 8 bits of data. For instance, a common automotive EPROM is referred to as a "256k × 8 EPROM". This EPROM contains 256-Kbytes 8-bit memory locations or 2,097,152 bits of information.

11.2.1 On-Chip Memory

On-chip microcontroller memory consists of some mix of five basic types: random access memory (RAM), read-only memory (ROM), erasable ROM (EPROM), electrically erasable ROM (EPROM), and flash memory. RAM is typically utilized for run-time variable storage and SFRs. The various types of ROM are generally used for code storage and fixed data tables.

The advantages of on-chip memory are numerous, especially for automotive applications, which are very size and cost conscious. Utilizing on-chip memory eliminates the need for external memory and the "glue" logic necessary to implement an address/data bus system. External memory systems are also notorious generators of switching noise and RFI due to their high clock rates and fast switching times. Providing sufficient on-chip memory helps to greatly reduce these concerns.

RAM. RAM may be defined as memory that has both read and write capabilities so that the stored information can be retrieved (read) and changed by applying new information to the cell (write). RAM found on microcontrollers is that of the static type that uses transistor cells connected as flip-flops. A typical six-transistor CMOS RAM cell is shown in Fig. 11.20. It consists of two cross-coupled CMOS inverters to store the data and two transmission gates, which provide the data path into or out of the cell. The most significant characteristic of static memory is that it loses its memory contents once power is removed. After power is removed, and once it is reapplied, static microcontroller RAM locations will revert to their default state of a logic "0". Because of the number of transistors used to construct a single cell, RAM memory is typically larger per bit than EPROM or ROM memory.

Although code typically cannot be executed from register RAM, a special type of RAM often referred to as *code RAM* is useful for downloading small segments of executable code. The difference between code and register RAM is that code RAM can be accessed via the

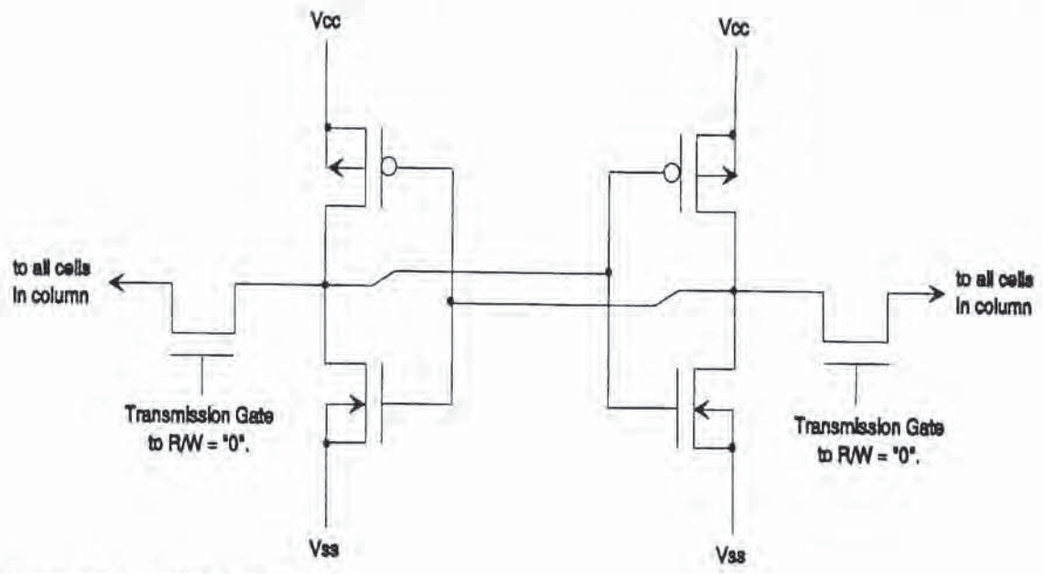


FIGURE 11.20 CMOS RAM memory cell.

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

memory controller, thus allowing code to be executed from it. Code RAM is especially useful for end-of-line testing during ECU manufacturing by allowing test code to be downloaded via the serial port peripheral.

ROM. Read-only memory (ROM), as the name implies, is memory that can be read but not written to. ROM is used for storage of user code or data that does not change since it is a nonvolatile memory that retains its contents after power is removed. Code or data is either entered during the manufacturing process (masked ROM, or MROM) or by later programming (programmable ROM, or PROM); either way, once entered it is unalterable.

A ROM cell by itself (Fig. 11.21) is nothing more than a transistor. ROM cells must be used in a matrix of word and bit lines (as shown in Fig. 11.22) in order to store information. The word lines are connected to the address decoder and the bit lines are connected to output buffers. The user's code is permanently stored by including or omitting individual cells at word and bit line junctions within the ROM array. For MROMs, this is done during wafer fabrication. For PROMs, this is done by blowing a fuse in the source/drain connection of each cell. To read an address within the array, the address decoder applies the address to the memory matrix. For any given intersection of a word and bit line, the absence of a cell transistor allows no current to flow and causes the transistor to be off. This indicates an unprogrammed ROM cell. The presence of a complete cell conducts and is sensed as a logical "0", indicating a programmed cell. The stored data on the bit lines is then driven to the output buffers.

MROMs are typically used for applications whose code is stable and in volume production. After the development process is complete and the user's program has been verified, the user submits the ROM code to the microcontroller manufacturer. The microcontroller manufacturer then produces a mask that is used during manufacturing to permanently embed the program within the microcontroller. This mask layer either enables or disables individual ROM cells at the junctions of the word and bit lines. An advantage of MROM microcontrollers is that they come with user code embedded, which saves time and money since postproduction programming is not necessary. A disadvantage of MROM devices is that, since the mask with the user code has to be supplied early in the manufacturing process, throughput time (TPT) is longer.

Some versions of ROM (such as Intel's Quick-ROM) are actually not ROMs, but rather EPROMs, which are programmed at the factory. These devices are packaged in plastic devices, which prevents them from being erased since ultraviolet light cannot be applied to the actual EPROM array. Throughput time for QROMs is faster since the user code isn't required until after the actual manufacturing of the microcontroller is complete. As with

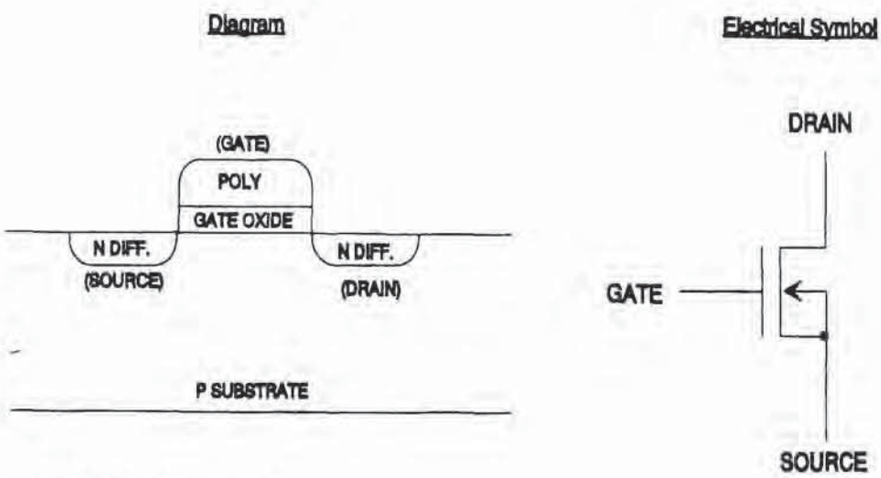


FIGURE 11.21 ROM memory cell.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

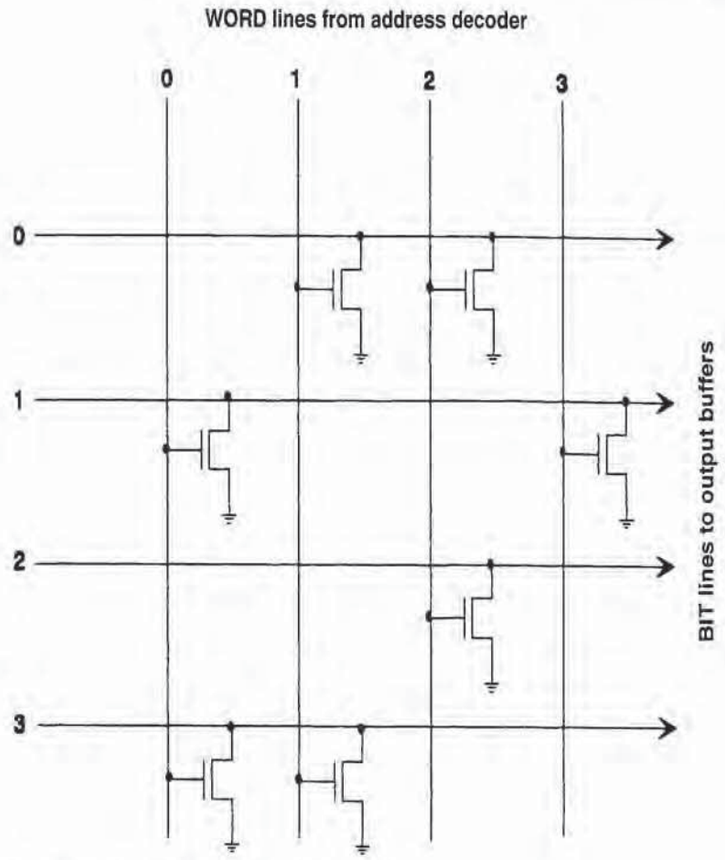


FIGURE 11.22 Simplified ROM memory matrix.

MROMs, the user supplies the ROM code to the microcontroller manufacturer. Instead of creating a mask with the ROM code, the manufacturer programs it into the device just prior to final test.

EPROM. EPROM devices are typically used during application development since this is when user code is changed often. EPROMs are delivered to the user unprogrammed. This allows the user to program the code into memory just prior to installation into an ECU module. Many EPROM microcontrollers actually provide a mechanism for in-module programming. This feature allows the user to program the device via the serial port while it is installed in the module. EPROM devices come assembled in packages either with or without a transparent window. Windowed devices are true EPROM devices that allow the user to erase the memory contents by exposing the EPROM array to ultraviolet light. These devices may be reprogrammed over and over again and thus are ideally suited for system development and debug during which code is changed often. EPROM devices assembled in a package without a window are commonly referred to as *one-time programmable devices* or OTPs. OTPs may only be programmed once, since the absence of a transparent window prevents UV erasure. OTPs are suited for limited production validation (PV) builds in which the code will not be erased.

A typical EPROM cell is shown in Fig. 11.23. It is basically an N-channel transistor that has an added poly1 floating gate to store charge. This floating gate is not connected and is surrounded by insulating oxide that prevents electron flow. The mechanism used to program an EPROM cell is known as *hot electron injection.* Hot electron injection occurs when very high drain (9-V) and select gate (12-V) voltages are applied. This gives the negatively charged electrons enough energy to surmount the oxide barrier and allows them to be stored on the gate.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

This has the same effect as a negative applied gate voltage and turns the transistor off. When the cell is unprogrammed, it can be turned on like a normal transistor by applying 5 V to the poly2 select gate. When it is programmed, the 5 V will not turn on the cell. The state of the cell is determined by attempting to turn on the cell and detecting if it turns on. Erasure is performed through the application of ultraviolet (UV) light, which gives just the right amount of energy necessary for negatively charged electrons to surmount the oxide barrier and leave the floating gate.

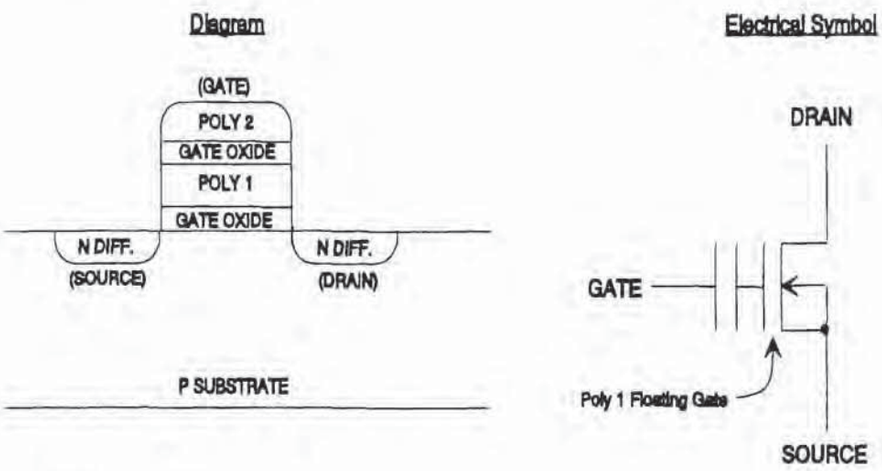


FIGURE 11.23 EPROM memory cell.

Flash. Flash memory is the newest nonvolatile memory technology and is very similar to EPROM. The key difference is that flash memory can be electrically erased. Once programmed, flash memory contents remain intact until an erase cycle is initiated via software. Like EEPROM, flash memory requires a programming and erase voltage of approximately 12.0 V. Since a clean, regulated 12-V reference is not readily available in automotive environments, this need is often provided for through the incorporation of an on-chip charge pump. The charge pump produces the voltage and current necessary for programming and erasure from the standard 5-V supply voltage. The advantage of flash is in its capability to be programmed *and* erased in-module without having to be removed. In-module reprogrammability is desirable since in-vehicle validation testing doesn't always allow for easy access to the microcontroller. Flash also allows for last-minute code changes, data table upgrades, and general code customization during ECU assembly. Since a flash cell is nearly identical in size to that of an EPROM cell, the high reliability and high device density capable with EPROM is retained. The main disadvantage of flash is the need for an on-chip charge pump and special program and erase circuitry, which adds cost.

A flash memory cell is essentially the same as an EPROM cell, with the exception of the floating gate. The difference is a thin oxide layer which allows the cell to be electrically erased. The mechanism used to erase data is known as *Fowler-Nordheim tunneling,* which allows the charge to be transferred from the floating gate when a large enough field is created. Hot electron injection is the mechanism used to program a cell, exactly as is done with EPROM cells. When the floating gate is positively charged, the cell will read a "1", when negatively charged, the cell will read a "0".

EEPROM. EEPROM (electrically erasable and programmable ROM, commonly referred to as E^2ROM) is a ROM that can be electrically erased and programmed. Once programmed, EEPROM contents remain intact until an erase cycle is initiated via software. Like flash, programming and erase voltages of approximately 12 V are required. Since a clean, regulated 12-V reference is not readily available in automotive environments, this requirement is satisfied using an on-chip charge pump as is done for flash memory arrays. Like flash, the advantage of EEPROM is its

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

capability to be programmed and erased in-module. This allows the user to erase and program the device in the module without having to remove it. EEPROM's most significant disadvantage is the need for an on-chip charge pump. Special program and erase circuitry also adds cost.

An EEPROM cell is essentially the same as an EPROM cell with the exception of the floating gate being isolated by a thin oxide layer. The main difference from flash is that Fowler-Nordheim electron tunneling is used for *both* programming and erasure. This mechanism allows charge to be transferred to or from the floating gate (depending upon the polarity of the field) when a large enough field is created. When the floating gate is positively charged, the cell will read a "1"; when negatively charged, the cell will read a "0".

11.2.2 Off-Chip Memory

Off-chip memory offers the most flexibility to the system designer, but at a price; it takes up additional PCB real estate as well as additional I/O pins. In cost- and size-conscience applications, such as automotive ABS, system designers almost exclusively use on-chip memory. However, when memory requirements grow to sizes in excess of what is offered on-chip (such as is common in electronic engine control), the system designer must implement an off-chip memory system. Off-chip memory is flexible because the user can implement various memory devices in the configuration of his choice. Most microcontrollers on the market today offer a wide variety of control pins and timing modes to allow the system designer flexibility when interfacing to a wide range of external memory systems.

Accessing External Memory. If circuit designers must use external memory in their applications, the type of external address/data bus incorporated onto the microcontroller should be considered. If external memory is not used, this will have, if any, impact upon the application. There are two basic types of interfaces used in external memory systems. Both of these are parallel interfaces in which bits of data are moved in a parallel fashion and are referred to as *multiplexed* and *demultiplexed* address/data buses.

Multiplexed Address/Data Buses. As the name implies, multiplexed address/data buses allow the address as well as the data to be passed over the same microcontroller pins by multiplexing the two in time. Figure 11.24 illustrates a typical multiplexed 16-bit address/data bus system as is implemented with Intel's 8XC196Kx family of microcontrollers.

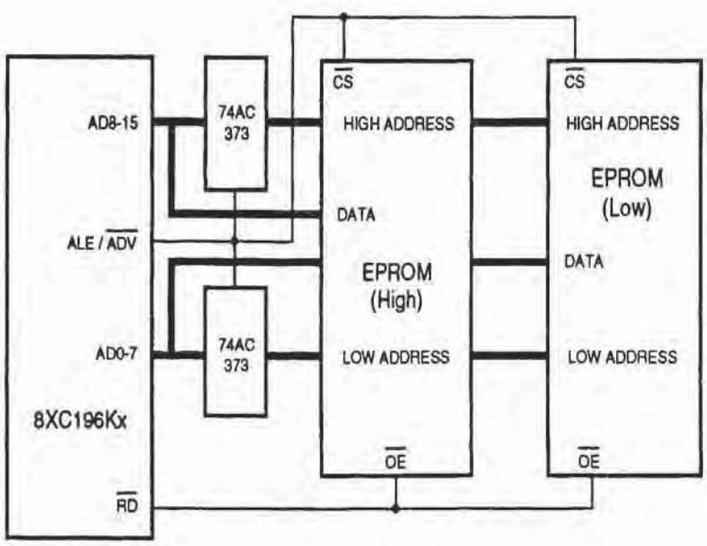


FIGURE 11.24 Multiplexed address/data bus system.

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

During a multiplexed bus cycle (refer to Fig. 11.25), the address is placed on the bus during the first half of the bus cycle and then latched by an external address data latch. The signal to latch the address comes from a signal generated by the microcontroller, called address latch enable (ALE). The address must be present on the bus for a specified amount of time prior to ALE being asserted. After the address is latched, the microcontroller asserts either a read (RD#) or a write (WR#) signal to the external memory device.

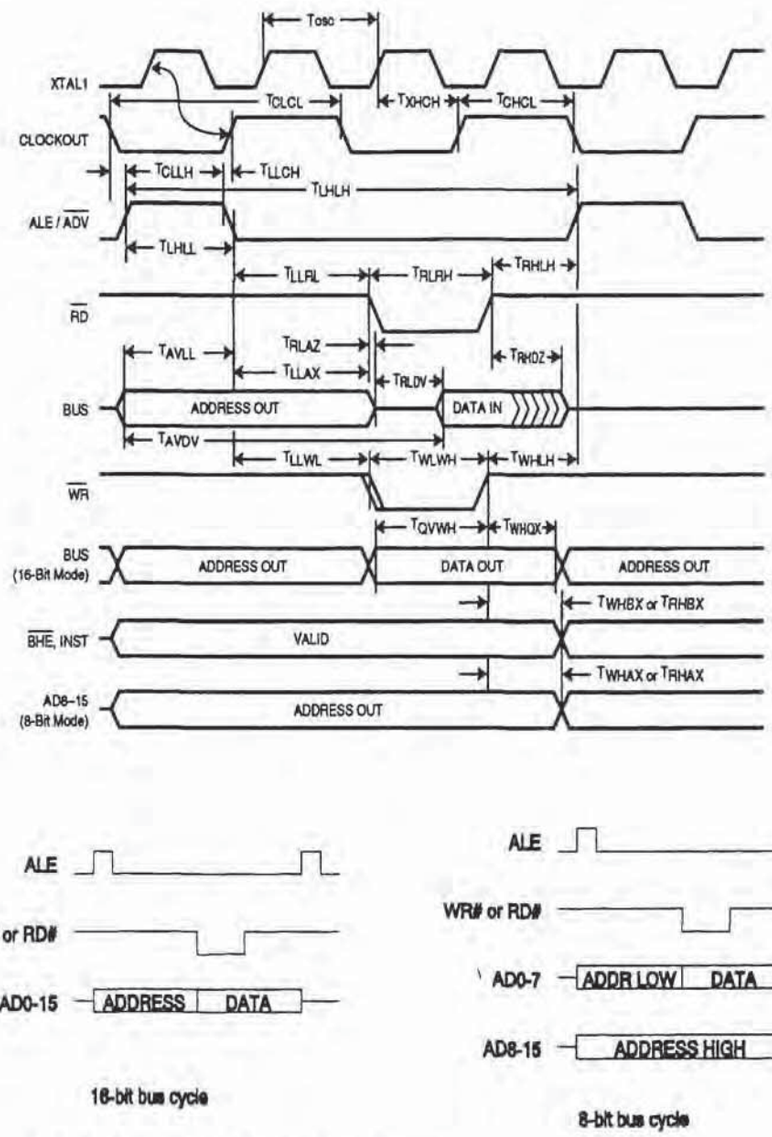


FIGURE 11.25 Multiplexed bus cycle and timing diagram.

For a read cycle, the microcontroller will pull its RD# output pin low and float the bus to allow the memory device to output the data located at the address latched on its address pins. The data returned from external memory must be on the bus and stable for a specified setup time before the rising edge or RD#, which is when the microcontroller latches the data.

For a write cycle, the microcontroller will pull its WR# pin low and then output data on the bus to be written to the external memory. After a specified setup time, the microcontroller will

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

release its WR# signal, which signals to the memory device to latch the data on the bus into the address location present on its address pins.

Advantages of multiplexed address/data bus systems are that fewer microcontroller pins are required since address and data share the same pins. For a true 16-bit system, this translates into a multiplexed system requiring 16 fewer pins (for address and data) than would be required by a demultiplexed system. A disadvantage is that an external latch is required to hold the address during the second half of the bus cycle; this adds to the component count.

Demultiplexed Address/Data Buses. Microcontrollers with demultiplexed address/data buses implement separate, dedicated address and data buses as shown in Fig. 11.26.

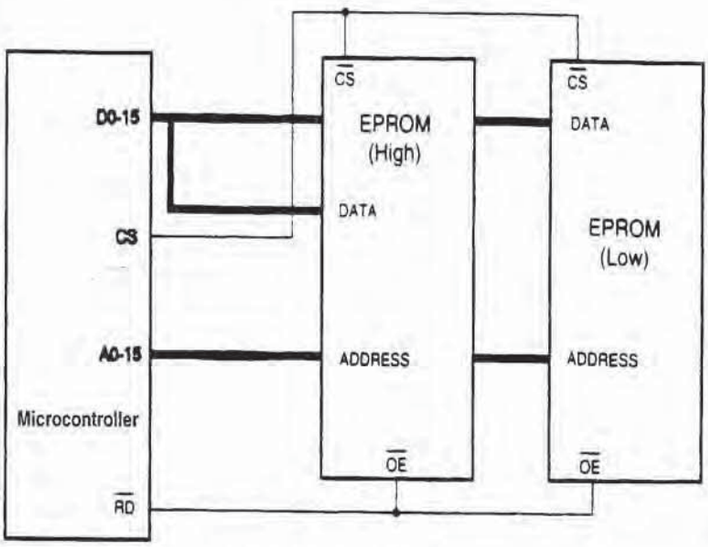


FIGURE 11.26 Typical demultiplexed address/data bus system.

The operation of a demultiplexed address/data bus is basically the same as the multiplexed type with the exception of not having an ALE signal to latch the address for the second half of the bus cycle. The operation of the RD#, WR#, address, and data lines is essentially the same as for that of a multiplexed system.

During a demultiplexed bus cycle, the microcontroller places the address on the address bus and holds it there for the entire bus cycle. For a read of external memory, the microcontroller asserts the RD# signal (or WR# for a write signal) just as would be done for a multiplexed bus cycle. The memory device will respond accordingly by either placing the data to be read on the data bus or by latching the data to be written off of the data bus. Figure 11.27 illustrates a simplified demultiplexed bus cycle.

An advantage of multiplexed address/data bus systems is that external data latches are not necessary, which saves on system component count. A disadvantage, as mentioned earlier, is that more microcontroller pins must be allocated for the interface, which leaves fewer pins for other I/O purposes.

11.3 LOW-SPEED INPUT/OUTPUT PORTS

Low-speed input/output (LSIO) ports allow the microcontroller to read input signals as well as provide output signals to and from other electronic components such as sensors, power drivers,

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

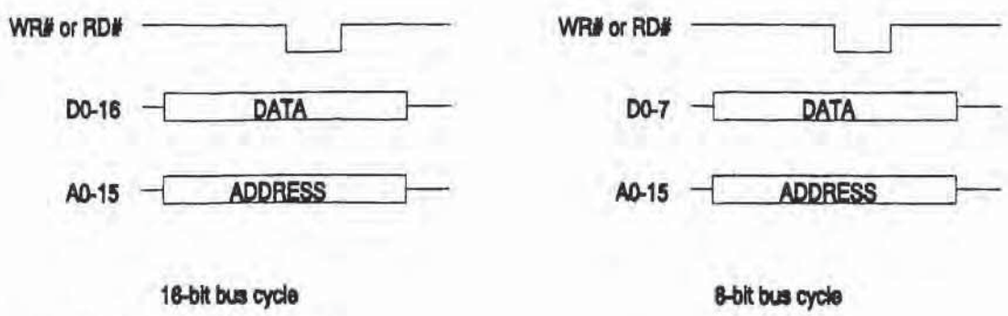


FIGURE 11.27 Demultiplexed bus cycle.

digital devices, actuators, and other microcontrollers. The term "low-speed" is used to describe these ports because unlike high-speed I/O (HSIO) ports which are interrupt driven, LSIO port data must be manually read and written by the user program. Interrupt-driven I/O is typically not possible on port pins configured for LSIO operation. It is common for modern high-performance microcontrollers to utilize multifunctional port pins which can be configured for a special function as well as LSIO. LSIO ports most commonly consist of eight port pins in parallel, which are supported by byte registers. For example, by writing to a single-byte special function register, an entire port can be configured, read, or written. Manipulating individual bits in the port register allows the user flexibility in accessing either single or multiple port pins.

11.3.1 Push-Pull Port Pin Configuration

The term *push-pull*, or *complementary*, output is commonly used to define a port pin that has the capability to output either a logic "1" or "0". Figure 11.28 shows a basic push-pull port pin configuration. Referring to Fig. 11.28, writing a "1" to the data output register enables the P-channel MOSFET and pulls the pin to +5 V, thus driving a logic "1" at the port pin. When a "0" is written

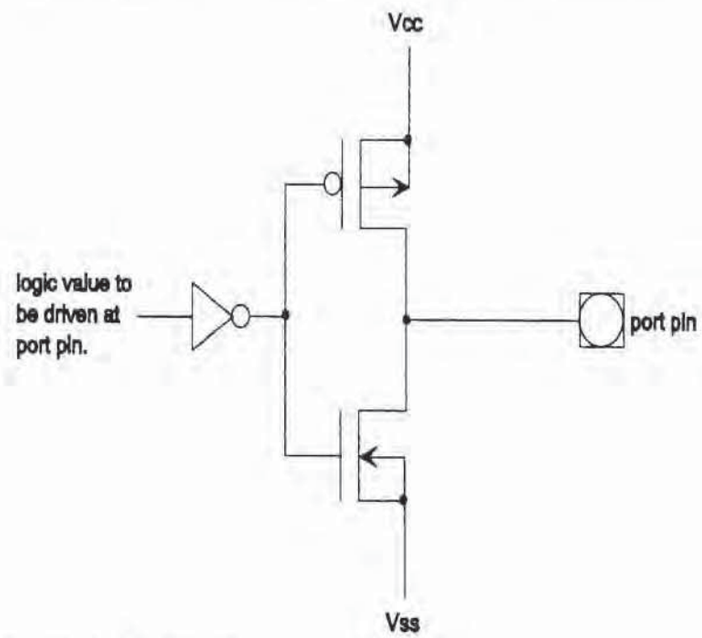


FIGURE 11.28 Push-pull port pin.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

to the data register, the N-channel MOSFET is enabled and thus provides a current path to ground which results in a logic 0 at the port pin. Note that during this time the P-channel pull-up MOSFET is disabled to prevent contention at the port pin. Also note that the port logic design does not allow both the P-channel and the N-channel devices to be driving at the same time.

11.3.2 Open-Drain Port Pin Configuration

Open-drain port pins (Fig. 11.29) are useful for handshaking signals over which multiple devices will have control. The fact that the P-channel transistor is either omitted or disabled dictates the need for an external pull-up resistor. An example of an application for open-drain port pins would be for a bus contention line between two microcontrollers communicating on a common bus. During normal operation, the line is pulled high by the external pull-up resistor to signal to either microcontroller that no contention exists. If one of the microcontrollers should detect contention on the bus, it simply outputs a logic "0", which signals the contention to the other processor. To output the "0", the port only has to overcome the external pull-up which the user should appropriately size to match the port drive specifications.

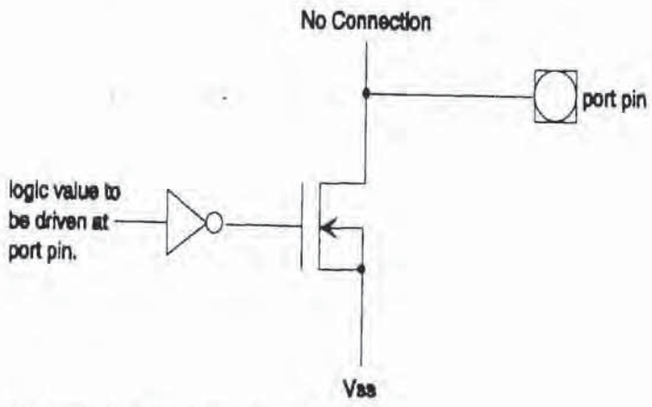


FIGURE 11.29 Open-drain port pin.

11.3.3 High-Impedance Input Port Pin Configuration

High impedance, or "Hi-z," port pins (Fig. 11.30) are used strictly as inputs since no drivers exist on these types of pins. Hi-z refers to the relatively high input impedance of the port pin. This high input impedance prevents the port pin circuitry from actively loading the input signal. Note that the pin is connected to the gates of a CMOS inverter, which drives internal circuitry. Usually a certain amount of hysteresis is built into these pins and is specified in the data sheet.

11.3.4 Quasi Bidirectional Port Pin Configuration

Quasi bidirectional (QBD) port pins are those that can be used as either input or output without the need for direction control logic. QBD port pins can output a strong low value or a weak high value. The weak high value can be externally overridden, providing an input function. Figure 11.31 shows a QBD port pin diagram and its transfer characteristic.

Writing a "1" to the port pin disables the strong low driver (Q2) and enables a very weak high driver (Q3). To get the pin to transition high quickly, a strong high driver (Q1) is enabled for one state time and then disabled (leaving only Q3 active).

It is important to keep in mind that since the port pin can be externally overridden with a logic "0", reading the port pin could falsely indicate that it was written as a logic "0".

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

The ability to overdrive the weak output driver is what gives the quasi bidirectional port pin its input capability. To reduce the amount of current that flows when the pin is externally pulled low, the weak output driver (Q4) is turned off when a valid logic "0" is detected. The input transfer characteristic of a quasi bidirectional port pin is shown in Fig. 11.31.

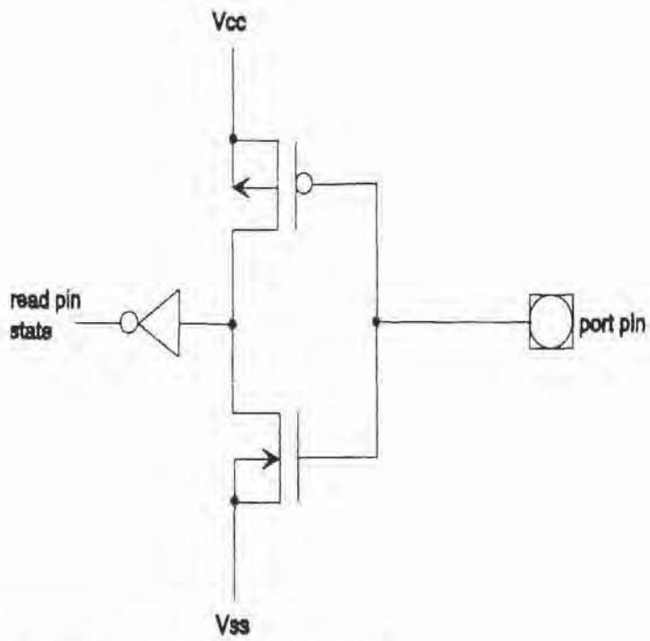


FIGURE 11.30 High-impedance input port pin.

11.3.5 Bidirectional Port Example

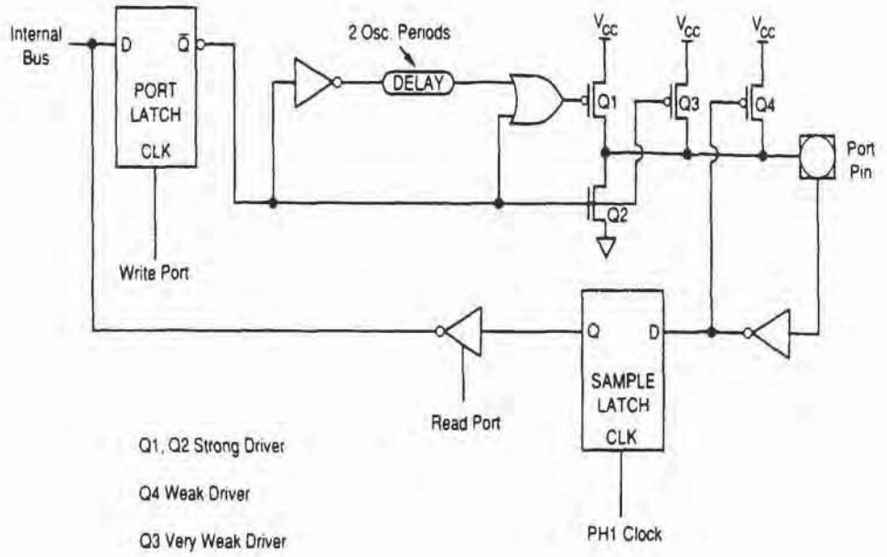
The following example describes the operation of a state-of-the-art bidirectional port structure. This particular structure is used upon newer members of Intel's MCS-96 automotive microcontroller family. A single port consists of eight multifunction, parallel port pins (see Fig. 11.32), which are controlled (on a by-pin basis) with four special function registers referred to as Px_PIN, Px_REG, Px_MODE, and Px_DIR. As is common with other high-performance microcontrollers, the pins of this port are shared with alternate special functions controlled by other on-chip peripherals. The Px_MODE register allows the programmer to choose either LSIO or the associated special function for any given port pin. Writing a "1" to the appropriate bit selects the corresponding pin as special function whereas a "0" selects LSIO. The function of the Px_PIN and Px_REG registers is fairly straightforward. In order to read the value on the pin, the user simply reads the Px_PIN register. To write a value to the Px_REG register, the user simply writes the desired output value to the Px_REG register. The Px_DIR register allows the user to configure the port pin as either input or output.

In order to prevent an undefined pin state during reset, port pins revert to a default state during reset. For the Intel Kx bidirectional port structure, this state is defined as a weak logic "1". The transistor that drives this state is labeled as WKPU in Fig. 11.32 and is asserted in reset until the user writes to the Px_MODE register to configure the port pin.

Ports such as this offer the user much flexibility in assigning their function within an application. Following are three examples that depict how these ports may be configured by the user by writing values to the appropriate bit within the port SFR. Also note that the eight pins of a port may be configured individually on a pin-by-pin basis.

To configure a given port pin as a high-impedance input pin, the user must write the following values to the corresponding bit within the port SFR.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540



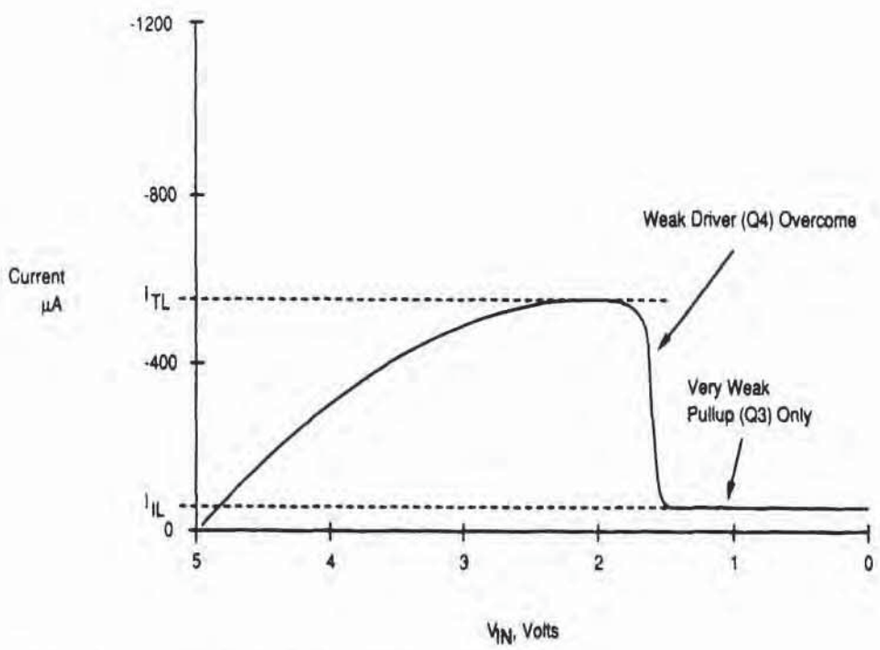


FIGURE 11.31 Quasi bidirectional port pin and transfer characteristic.

Px_MODE: "0" selects the pin as LSIO and disables weak pull-up.
Px_DIR: "1" disables operation of the N-channel transistor.
Px_REG: "1" disables the N-channel transistor.

To configure a given port pin for push-pull operation, the following values must be written to the corresponding bit within the port SFR.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

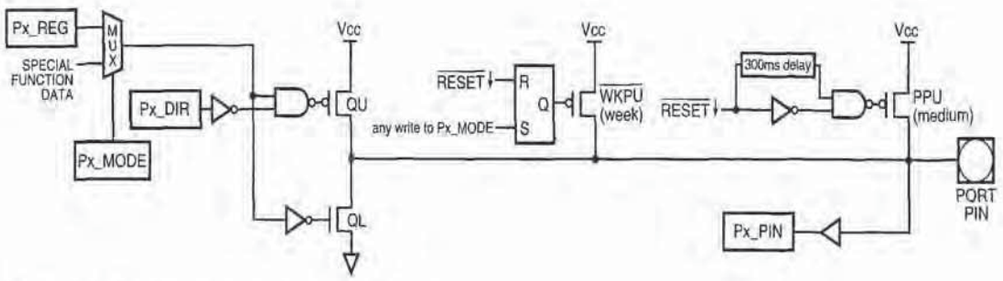


FIGURE 11.32 Bidirectional port structure example.

Px_MODE: "0" selects the pin as LSIO and disables weak pull-up transistor.

Px_DIR: "0" enables operation of both the N- and P-channel transistors.

Px_REG: "0" or "1" drives that value at the port pin.

To configure a port pin for open-drain operation, the user must write the following values to the corresponding bits within the port SFR.

Px_MODE: "0" selects the pin as LSIO and disables weak pull-up transistor.

Px_DIR: "1" disables operation of the N-channel transistor.

Px_REG: "1" disables the P-channel transistor / achieves Hi-Z state.
"0" enables the N-channel transistor / drives "0" at pin.

11.4 HIGH-SPEED I/O PORTS

Perhaps the most demanding of automotive microcontroller applications is electronic engine control and antilock braking/traction control. These applications both require the microcontroller to detect, process, and respond to external signals or "events" within relatively short periods of time. Sometimes referred to as a capture/compare module, a microcontroller's HSIO (high-speed input/output) peripheral allows the microcontroller to capture an event as it occurs. The term *capture* refers to a series of events that begins with the microcontroller detecting a rising or falling edge upon a high-speed input pin. At the precise moment this edge is detected, the value of a software timer is loaded into a time register and an interrupt is triggered. This gives the microcontroller the relative time at which the event occurred. An HSIO peripheral also provides compare functions by detecting an internal event, such as a timer reaching a particular count value. When the particular count value is detected, the HSIO unit will generate a specified event (rising or falling edge) on a port pin. This feature is ideal for generating PWM waveforms or synchronizing external events with internal events.

For example, consider a typical ABS microcontroller which must detect, capture, and calculate wheel speeds; respond with signals to hydraulic solenoids; and perform many other background tasks all within a loop time of about 5 ms. The wheel speed signals are input to the microcontroller as square waves with frequencies up to 7000 Hz (approximately one edge every 71 μs). The microcontroller must have the performance necessary to capture and process these edges on as many as four wheel speed inputs. HSIO peripherals, along with the interrupt structure, play a major role in the microcontroller's ability to perform this function.

Nearly every microcontroller manufacturer has its own proprietary HSIO peripheral. For purposes of this section, the event processor array (EPA) HSIO peripheral, which is used by Intel's 87C196KT automotive microcontroller, will be discussed.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

11.4.1 High-Speed Input and Output Peripheral

High-speed input/output peripherals typically consist of a given number of capture/compare modules, a timer/counter structure, control and status SFRs, and an interrupt structure of some type. Figure 11.33 shows a block diagram of the EPA peripheral. The main components of the EPA are ten capture/compare channels, two compare only channels, and two timer/counters. The capture/compare channels are configured independently of each other. The two timer/counters are shared between the various capture/compare channels. Each capture/compare channel has its own dedicated SFR's: EPAx_TIME and EPAx_CON (x designates the channel number).

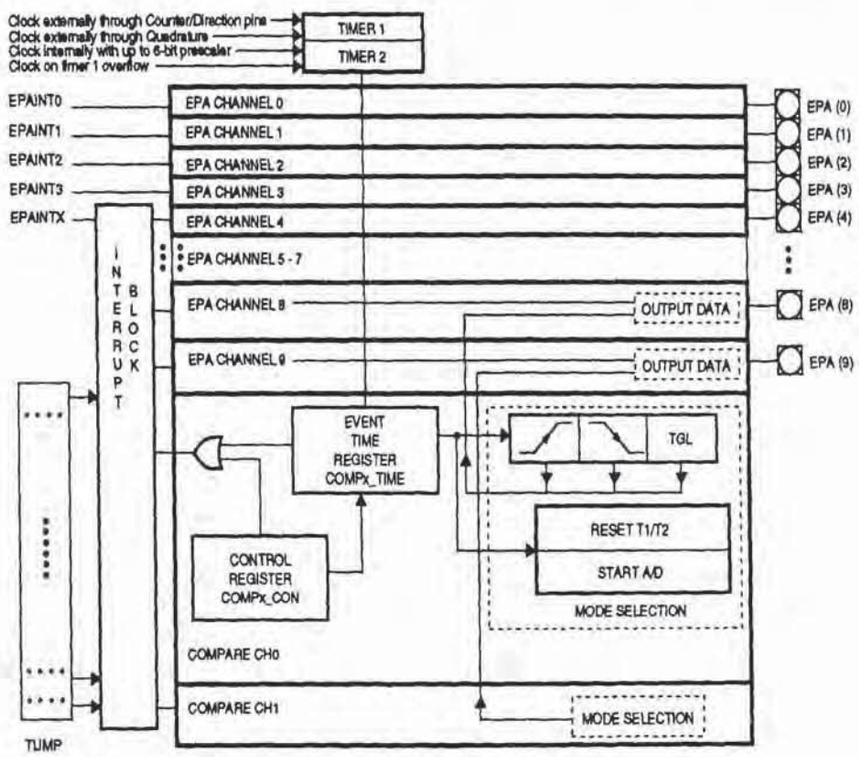


FIGURE 11.33 Example HSIO peripheral: Intel's EPA peripheral.

11.4.2 Timer/Counter Structures

High-performance microcontrollers typically integrate one or more timer/counters onto their silicon. A microcontroller's timer/counter structure provides a time base to which all HSIO events are referenced. Timers are clocked internally, whereas counters are clocked from an external clock source. Timers are often very flexible structures, in which programmers have the capability to configure the timer/counters to meet their application's particular needs. The 87C196KT has two 16-bit timer/counters referred to as TIMER1 and TIMER2. As 16-bit timer/counters, each timer has the capability of counting to 2^{16} or 65,536 before overflowing. The user has the option of triggering an interrupt upon overflow of a timer/counter. Each of these two timers can be independently configured using the TxCONTROL SFR as shown in Fig. 11.34, where x specifies either 1 or 2 for Timer1 or Timer2, respectively.

Bits number 3, 4, and 5 are the mode bits that allow the user to configure the clocking source and direction of each timer/counter. The clock rate can be based either upon the fre-

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

TxCONTROL SFR

7	6	5	4	3	2	1	0
CE	UD	M2	M1	M0	P2	P1	P0

CE: Count Enable: "0" = disable timer, "1" = enable timer

UD: Up/Down: "0" = count up, "1" = count down

MODE:	M2, M1, M0	Clock source	Direction determined by:
	0 0 0	XTAL/4	state of UD bit
	0 0 1	TxCLK pin	state of UD bit
	0 1 0	XTAL/4	state of TxDIR pin
	0 1 1	TxCLK pin	state of TxDIR pin
	1 0 0	Timer1 overflow	state of UD bit
	1 1 0	Timer1 overflow	same as Timer1
	1 1 1	Quadrature clocking using TxCLK and TxDIR pins	

Prescale:	P2, P1, P0	Clock prescale values
	0 0 0	÷ by 1 (250 ns @ 16 MHz xtal frequency)
	0 0 1	÷ by 2 (500 ns @ 16 MHz xtal frequency)
	0 1 0	÷ by 4 (1 µs @ 16 MHz xtal frequency)
	0 1 1	÷ by 8 (2 µs @ 16 MHz xtal frequency)
	1 0 0	÷ by 16 (4 µs @ 16 MHz xtal frequency)
	1 0 1	÷ by 32 (8 µs @ 16 MHz xtal frequency)
	1 1 0	÷ by 64 (16 µs @ 16 MHz xtal frequency)
	1 1 1	reserved

FIGURE 11.34 Timer control SFR example.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

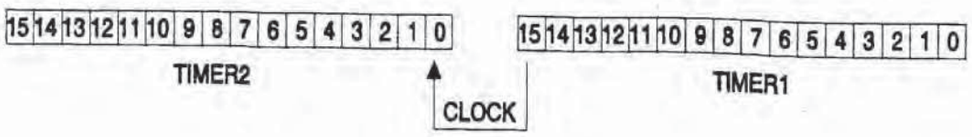


FIGURE 11.35 Cascading of timer/counters.

quency that the microcontroller is being clocked at the XTAL pins or upon the input frequency on another pin referred to as TxCLK. The user also has the option of either having the logic level of another pin (TxDIR) or the UD bits in TxCONTROL determine the direction (up/down) that the timer/counter is clocked.

For those applications that require a 32-bit timer/counter, the user has the option (using the mode bits) to direct the overflow of TIMER1 to clock TIMER2. This is known as cascading and essentially creates a 32-bit timer/counter as shown in Fig. 11.35.

11.4.3 Input Capture

Input capture refers to the process of capturing a current timer value when a specific type of event occurs. An excellent example of high-speed input capture can be illustrated with a basic automotive ABS input capture algorithm that calculates the frequency of a wheel speed input. The signals from the wheel speed sensors are input into the microcontroller's EPA pins as square waves. Consider the generic wheel speed input capture example shown in Fig. 11.36.

Two timers (1 and 2) are used in this example. Timer1 is used in conjunction with an EPA channel to provide a 5-ms software timer (this is a compare function that will be discussed in the next section). The 5 ms is the main loop time used in generic ABS algorithms. Timer2 is used in conjunction with one or more EPA channels to capture the relative times at which edges occur on wheel speed inputs. The EPA is configured to capture falling edges and initiate an interrupt, which stores the event time and increments an edge count. To simplify this example, we will consider only a single input channel.

The process starts by EPA interrupts being enabled after Timer1 starts a new 5-ms timer count. The first falling edge causes an interrupt that stores the event time (T2) into a variable *initial time* and increments an edge count. The next edge causes an interrupt in which the event time (T2+x) is stored into a variable called *final time* and increments the edge count.

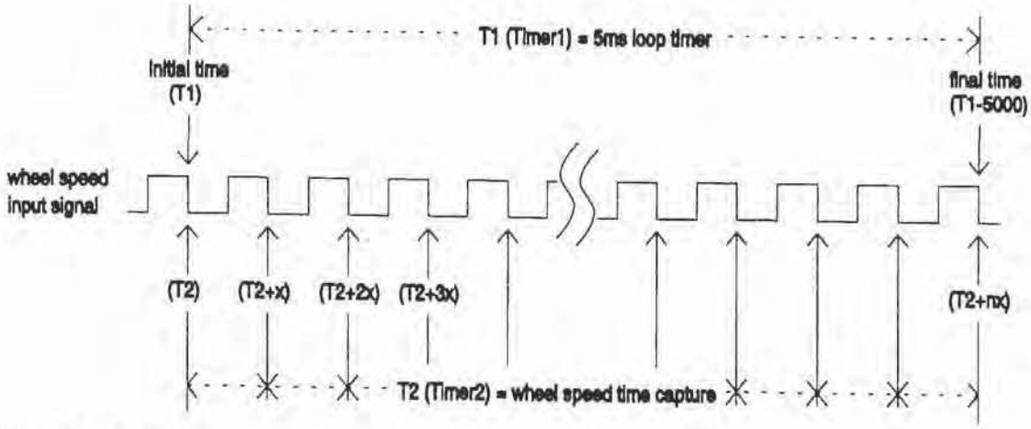


FIGURE 11.36 Input capture example using EPA peripheral.

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Subsequent edges' event times are also stored into *final time* until Timer1's 5-ms count expires. At this point, final time contains the time at which the last edge to occur was captured. The average period of the input waveform can then be calculated with the following equation:

$$\text{input period} = (\textit{final time} - \textit{initial time}) / \text{edge count}$$

11.4.4 Output Compare

Output compare refers to the process of generating an event when a timer value matches a predetermined time value. The event may be to generate an interrupt, toggle an output pin, perform an A/D conversion, and so forth. Following is an example that shows the steps necessary to generate an event every 50 μs:

1. Enable the output compare channel's interrupt.
2. Initialize the timer to count up at 1 μs per timer tick.
3. Initialize the output compare channel to re-enable and reset the timer (to zero) when a timer match occurs.
4. Initialize the output compare channel to produce the desired event when a timer match occurs.
5. Write 32h (50 decimal) to the appropriate output compare channel's time register.
6. Enable the timer to start the process.
7. A compare channel interrupt will be generated every 50 μs.

Since the example re-enables and zeros the timer, the event will occur continuously until the user's program halts the process.

Software Timers. Software timers such as the 5-ms timer used in the ABS wheel speed capture example can be set up easily using a compare channel and a timer. The following software timer procedure is very similar to that used in the previous output compare example:

1. Enable the compare channel's interrupt.
2. Initialize the timer to count up at 1 us per timer tick.
3. Initialize the output compare channel to re-enable and reset the timer (to zero) when a timer match occurs.
4. Initialize the output compare channel to produce an interrupt (5-ms ISR) when a timer match occurs.
5. Write 1388h (5000 decimal) to the appropriate output compare channel's time register.
6. Enable the timer to start the process.
7. An compare channel interrupt will be generated every 5 ms.

11.4.5 Pulse-Width Modulation (PWM)

Pulse-width modulation (PWM) peripherals provide the user with the ability to generate waveforms that have specified frequencies and duty cycles. PWM waveforms are typically used to generate pulsed waveforms used for motor control or they may be filtered to produce a smooth analog signal. HSIO peripherals typically provide for PWM waveform generation, although the methods are not usually as efficient as dedicated PWM peripherals. A basic example of creating a PWM waveform using an HSIO peripheral's output compare function is described in Sec. 11.4.4.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

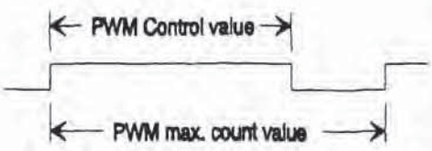


FIGURE 11.37 PWM waveform time values.

PWM Peripheral. The components of a basic automotive microcontroller's PWM peripheral include a counter (typically 8-bit), a comparator, a holding register, and a control register. The counter typically has a prescaler that allows the user to select the clock rate of the counter, which allows for selectable PWM frequencies. Without prescaling capability, an 8-bit counter would only allow for a period of 256 state times. The PWM control register determines how long the PWM output is held high during the pulse, effectively controlling the duty cycle as shown in Fig. 11.37. For an 8-bit PWM counter, the value written to the PWM control register can be from 0 to 255 (equating to 255 state times with no prescaling). Note that PWM peripherals do not typically allow for a 100 percent duty cycle because the output must be reset when the counter reaches zero.

The operation of a PWM peripheral is rather simple. The PWM control register's value (assume 8-bit for this example) is loaded into a holding register when the 8-bit counter overflows. The comparator compares the contents of the holding register to the counter value. When the counter value is equal to zero, the PWM output is driven high. It remains high until the counter value matches the value in the holding register, at which time the output is pulled low. When the counter overflows, the output is again switched high. Figure 11.38 shows typical PWM output waveforms.

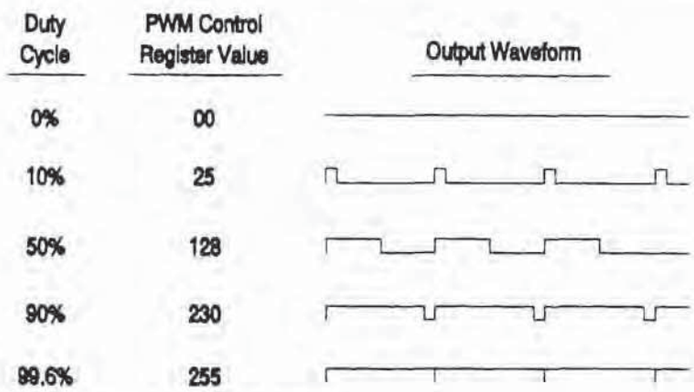


FIGURE 11.38 PWM output waveforms.

11.5 SERIAL COMMUNICATIONS

It is often necessary for automotive microcontrollers to have the capability to communicate with other devices both internal and external to the ECU. Within an ECU a microcontroller may have to communicate with other devices such as backup processors, shift registers, watchdog timers, and so forth. It is not uncommon for automotive microcontrollers to communicate with devices external to the ECU, such as other modules within the vehicle and even diagnostic computers at a service station. All of these communication examples require a large quantity of data to be transmitted/received in a short period of time. Also consider that this communication must utilize as few pins of the microcontroller as possible in order to save valuable PCB board space. These requirements all support the need for serial communications.

Serial communications provides for efficient transfer of data while utilizing a minimum number of pins. Serial communications is performed by transferring a group of data bits, one at a time, sequentially over a single data line. Each transmission of a group of bits (typically a

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

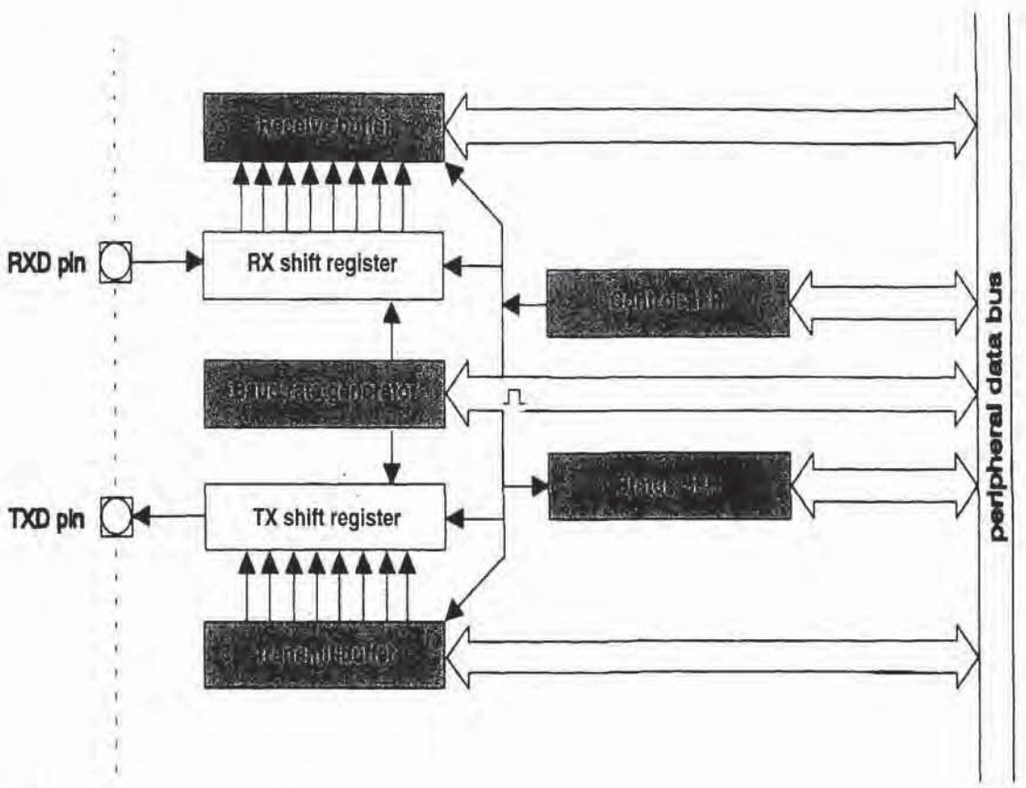


FIGURE 11.39 Serial port block diagram.

byte of data) is known as a data frame. This transfer of data takes place at a given speed, which is referred to as the baud rate and is typically specified in bits/second.

A typical microcontroller serial port consists of data buffers, data registers, and a baud rate generator. Interface to the outside world takes place via the transmit (TXD) and receive (RXD) pins. A block diagram for a typical serial port peripheral is shown in Fig. 11.39. By writing to the serial port control register, users are able to customize the operation of the serial port to their particular application's requirements.

The baud rate generator is used to provide the timing necessary for serial communications and determines the rate at which the bits are transmitted. In synchronous modes, the baud rate generator provides the timing reference used to create clock edges on the clock output pin. In asynchronous modes, the baud rate generator provides the timing reference used to latch data into the RX pin and clock it out of the TX pin.

11.5.1 Synchronous Serial Communications

Sometimes an application does not allow asynchronous serial communications to take place due to variations in clock frequency, which results in unacceptable baud rate error. Some applications simply require some sort of shift register I/O. Synchronous communication involves an additional clock pin, which is used to signal the other device that data being transferred are valid and ready to be read. Often when the user configures the serial port to work in a synchronous mode, the TXD pin automatically reverts to supplying the clock and the RXD pin automatically becomes the data pin. This configuration prevents an additional pin from having to be reserved for use as a serial clock pin. When a synchronous data transfer is initiated, a series of eight clock pulses is emitted from the clock pin at a predetermined baud rate as shown in Fig. 11.40.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

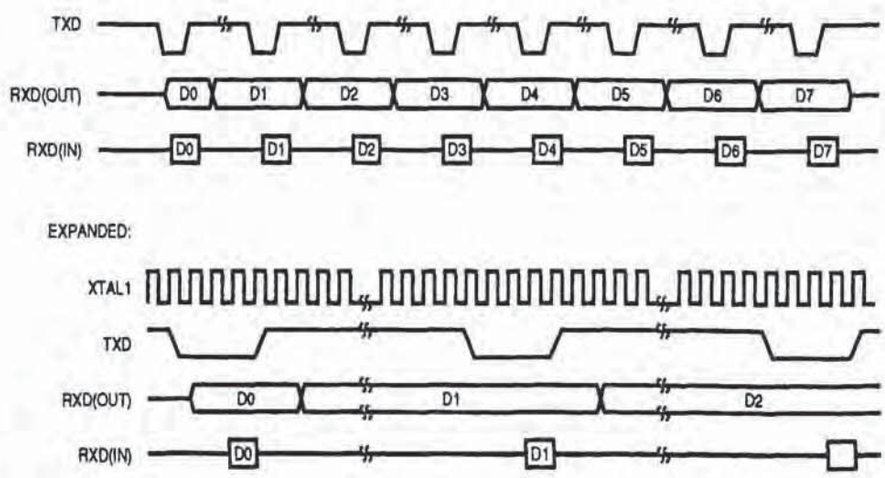


FIGURE 11.40 Synchronous serial mode data frame.

An example of synchronous serial communications is shown in Fig. 11.41. Assume that processor A is to transfer a byte of data to processor B. The program executing in processor A initiates a serial transmission by writing the data byte to be transmitted into the transmit buffer. Assuming microcontroller A's serial port is enabled for transmission, writing to the transmit buffer results in a series of eight clock pulses to be emitted from microcontroller A's clock pin. The first falling edge of the clock will signal to processor B that bit 0 (LSB) is ready to be read into its receive buffer. Microcontroller A will place the next data bit on the TXD pin with each rising clock edge. With B's serial port enabled for reception, each falling edge will result in another data bit being shifted into B's receive buffer. When B's receive buffer is full, the received data byte will be loaded into its receive register and will signal its CPU that the reception has been completed and the data is ready for use.

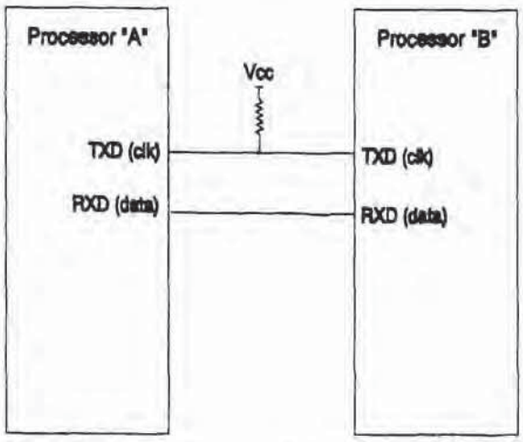


FIGURE 11.41 Synchronous serial communications example.

Shift Register Based I/O Expansion. A common application for synchronous serial transmission is shift register based I/O expansion as shown in Fig. 11.42. In this circuit, a 74HC164 8-bit serial-in/parallel-out shift register is used to provide eight parallel outputs with a single serial input. The 74HC165 8-bit parallel-in/serial out shift register shown provides a single serial input resulting from eight parallel input signals. This allows the system designer to

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

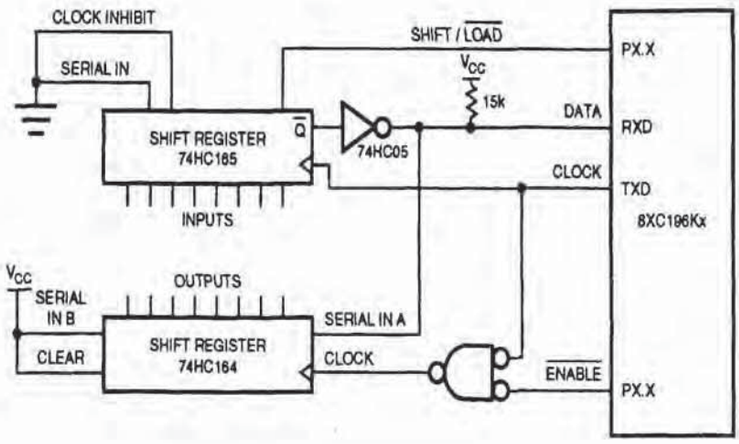


FIGURE 11.42 Shift register based I/O expansion example.

implement an additional 8-bit output port and additional 8-bit input port (16 signals total) using only four pins on the microcontroller. This expansion scheme allows a designer to achieve a greater number of I/O pins without having to upgrade to a microcontroller with a higher pin count.

To output data using this I/O expansion method, the user code simply writes a byte to the serial port transmit register to initiate data transfer. This causes the written byte to be shifted out of the microcontroller's RXD pin and into the 74HC164 one bit at a time. The data is reflected at the output pins of the 74HC164 as each bit is shifted in. For address/data bus emulation, another microcontroller pin may be utilized to indicate valid data to the intended receiving device.

To receive eight bits of data in parallel using this method, the user's code must latch the data on the 74HC165's input pins into its shift register by asserting the *shift/load* signal. After this is accomplished, the user's code simply needs to enable the serial port receive circuitry to receive the data one bit at a time into its receive buffer.

11.5.2 Asynchronous Serial Communications

The most common type of serial communications is asynchronous. As its name implies, asynchronous communication takes place between two devices without use of a clock line. Data is transmitted out the transmit buffer and received into the receive buffer independently at a speed determined by the baud rate generator. Most microcontrollers offer several modes of asynchronous serial communication.

Standard Asynchronous Mode. The standard asynchronous mode consists of 10 bits: a start bit, eight data bits (LSB first), and a stop bit, as shown in Fig. 11.43. After the user initiates a transmission, data is automatically transmitted from the TX pin at the specified baud rate.

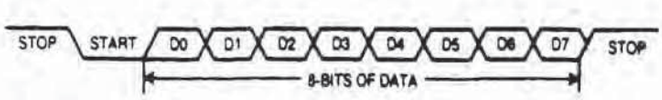


FIGURE 11.43 Standard asynchronous mode data frame.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

A parity function is also implemented, which provides for a simple method of error-detection. Data transmitted will consist of either an odd or even number of logical "1"s. If even parity is enabled, the parity bit will either be set to a "1" or a "0" to make the number of "1"s in the data byte even. If odd parity is enabled, the parity bit will be set to the appropriate value to make the number of "1"s in the data byte odd. For instance, consider the data byte 11010010b. If even parity is enabled, the parity bit will be set to a "0" since there is already an even number of "1"s. If odd parity were enabled, the parity bit would be set to a "1" since another "1" would be needed to provide an odd number of "1"s. If the parity function is enabled (usually through a serial port control register), the parity bit is sent instead of the eighth data bit and parity is checked on reception. The occurrence of parity errors is typically flagged in a serial port status register to alert the microcontroller to corrupted data in the receive register.

Multiprocessor Asynchronous Serial Communications Modes. Two other common serial communications modes which are used on automotive microcontrollers are the asynchronous 9th-bit recognition mode and the asynchronous 9th-bit mode. These two modes are commonly used together for multiprocessor communications where selective selection on a data link is required. Both modes are similar to the standard asynchronous mode with the exception of an additional ninth data bit in the data frame as shown in Fig. 11.44.

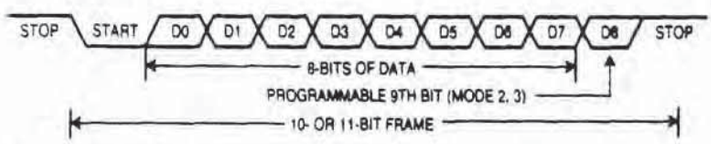


FIGURE 11.44 Asynchronous 9th-bit data frame.

The 9th-bit recognition mode consists of a start bit, nine data bits (LSB first), and a stop bit. For transmission, the ninth bit can be set to "1" by setting a corresponding bit in the serial port control register before writing to the transmit buffer. During reception, the receive interrupt bit is *not* set unless the ninth data bit being received is set to a logic "1".

The 9th-bit mode uses a data frame identical to that of the 9th-bit recognition mode. In this mode, a reception will always cause a receive interrupt, regardless of the state of the ninth data bit.

A multiprocessor data link is fairly simple to implement using these two modes. Microcontrollers within the system are connected as shown in Fig. 11.45. The master microcontroller is set to the 9th-bit recognition mode so that it is always interrupted by serial receptions. The slave microcontrollers are set to operate in the 9th-bit recognition mode so that they are interrupted on receptions only if the ninth data bit is set. Two types of data frames are used: address frames, which have the ninth bit set, and data frames, which have the ninth bit cleared. When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address frame which identifies the target slave. Slaves in the 9th-bit recognition mode are not interrupted by a data frame, but an address frame interrupts all slaves. Each slave can examine the received byte and see if it is being addressed. The addressed slave then switches to the 9th-bit mode to receive data frames, while the slaves that were not addressed stay in the 9th-bit recognition mode and continue without interruption.

11.6 ANALOG-TO-DIGITAL CONVERTER

Analog-to-digital converter (A/D) peripherals allow automotive microcontrollers to sense and assign digital values to analog input voltages with considerable accuracy. An analog input

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

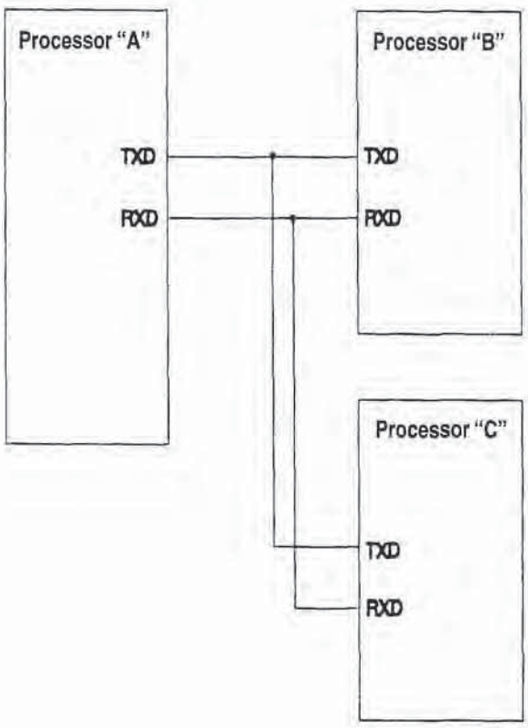


FIGURE 11.45 Asynchronous 9th-bit data frame.

may be defined as a voltage level that varies over a continuous range of values as opposed to the discrete values of digital signals.

11.6.1 Types of A/D Converters

The vast majority of A/D converters available on microcontrollers are of the successive approximation (S/A) type. Other types include flash A/D converters, in which conversions are completed in a parallel fashion and are performed at speeds measuring tens-of-nanoseconds. The drawback is that flash A/D converters require a great deal of die space when integrated on a microcontroller. It is because of their relatively large size that flash A/D converters are seldom offered on microcontrollers. Dual-slope A/D converters offer excellent A/D accuracy but typically take a relatively long period of time to complete a conversion. S/A A/D converters are very popular because they offer a compromise among accuracy, speed, and die-size requirements. The main drawback to successive approximation converters is that implementing the capacitor and resistor ladders takes a considerable amount of die space, although somewhat less than flash A/Ds. These converters are also somewhat susceptible to noise, although there are proven ways to reduce the effects of noise within a given application. The advantage of S/A converters is that they combine the best of other types of converters. They are relatively fast and do not take up excessive die space.

S/A converters typically consist of a resistor ladder, a sample capacitor, an input multiplexer, and a voltage comparator. A typical S/A converter is shown in Fig. 11.46. The resistor ladder is used to produce reference voltages for the input voltage comparison. A sample capacitor is utilized to capture the input voltage during a given period of time known as the sample time. Sample time can be defined as the amount of time that an A/D input voltage is applied to the sample capacitor.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

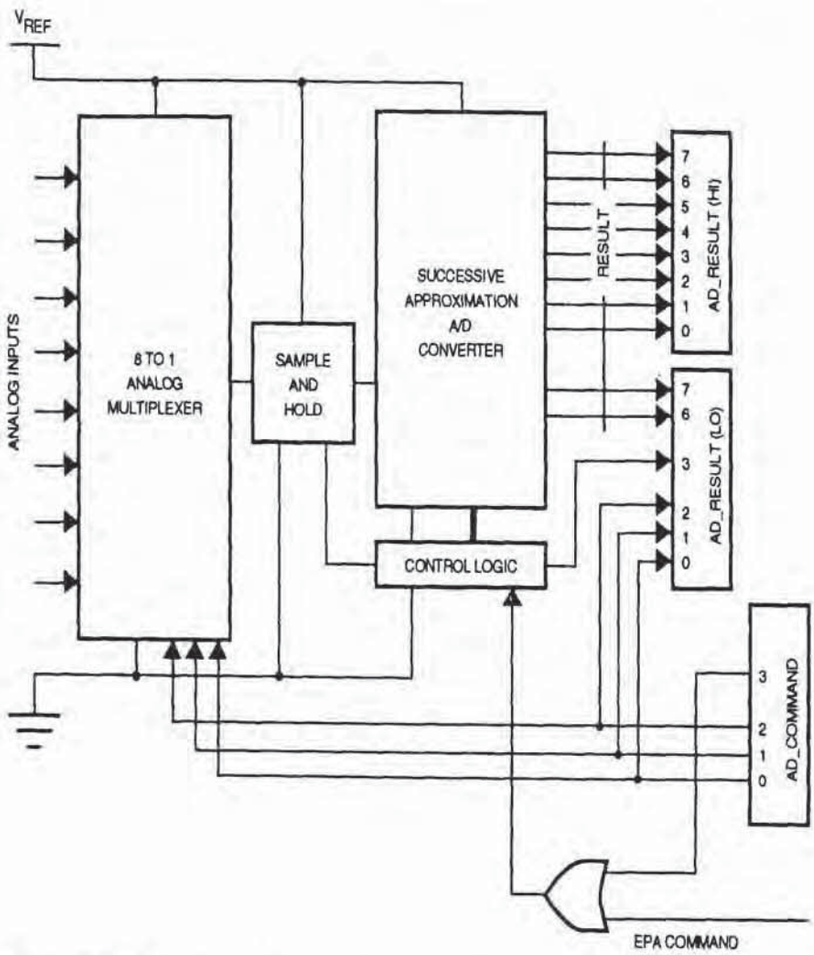


FIGURE 11.46 Typical successive approximation converter.

A successive approximation algorithm is used to perform the A/D conversion. A typical S/A converter consists of a 256-resistor ladder, a comparator, coupling capacitors, and a 10-bit successive approximation register (SAR), along with SFRs and logic to control the process. The resistor ladder provides 20-mV steps (with V_{ref} = 5.12 V), while capacitive coupling creates 5-mV steps within the 20-mV ladder voltages. Therefore, 1024 internal reference voltage levels are available for comparison against the analog input to generate a 10-bit conversion result. Eight-bit conversions use only the resistor ladder, providing 256 levels.

11.6.2 The A/D Conversion Process

The successive approximation conversion compares a reference voltage to the analog input voltage stored in the sampling capacitor. A binary search is performed for the reference voltage that most closely matches the input. The ½ full-scale reference voltage is the first tested. This corresponds to a 10-bit result in which the most significant bit is zero and all other bits are one (0111 1111 11b). If the analog input is less than the test voltage, bit 10 is left at zero and a new test voltage of ¼ full scale (0011 1111 11b) is tested. If this test voltage is less than the analog input voltage, bit 9 of the SAR is set and bit 8 is cleared for the next test (0101 1111

ENGR & MATH SCI LIBRARY UNLV

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

11b). This binary search continues until 8 or 10 tests have occurred, at which time the valid 8-bit or 10-bit result resides in the SAR where it can be read by software.

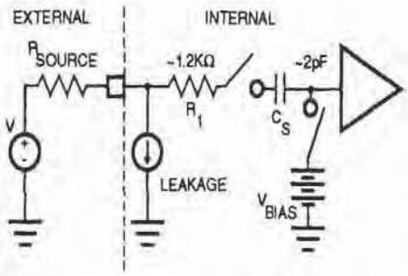


FIGURE 11.47 Idealized interface circuitry.

11.6.3 A/D Interfacing

The external interface circuitry to an analog input is highly dependent upon the application and can impact converter characteristics. Several important factors must be considered in the external interface design: input pin leakage, sample capacitor size, and multiplexer series resistance from the input pin to the sample capacitor. These factors are idealized in Fig. 11.47.

The following example is for a 1-μs sample time and a 10-bit conversion. The external input circuit must be able to charge a sample capacitor (C_S) through a series resistance (R_1) to an accurate voltage, given a dc leakage (I_L). For purposes of this example, assume C_S of 2 pf, R_1 of 1.2 kΩ, and I_L of 1 μA.

External circuits with source impedances of 1 kΩ or less can maintain an input voltage within a tolerance of about 0.2 LSB (1.0 kΩ × 1.0 μA = 1.0 mV) given the dc leakage. Source impedances above 5 kΩ can result in an external error of at least one LSB due to the voltage drop caused by the 1-μA leakage. In addition, source impedances above 25 kΩ may degrade converter accuracy because the internal sample capacitor will not charge completely during the sample time.

Typically, leakage is much lower than the maximum specification specified by the microcontroller manufacturer. Given typical leakage, source impedance may be increased substantially before a one-LSB error is apparent. However, a high source impedance may prevent the internal sample capacitor from fully charging during the sample window. This error can be calculated using the following formula:

$$\text{Error (LSBs)} = \left(e^{\frac{-T_{SAM}}{RC}} \right) \times 1024$$

where T_{SAM} = sample time, μs
$R = R_{SOURCE} + R_1$, Ω
$C = C_S$, μf

The effects of this error can be minimized by connecting an external capacitor C_{EXT} from the input pin to ANGND. The external signal will charge C_{EXT} to the source voltage. When the channel is sampled, a small portion of the charge stored in C_{EXT} will be transferred to the internal sample capacitor. The ratio of C_S to C_{EXT} causes the loss in accuracy. If C_{EXT} is .005 μf or greater, the maximum error will be –0.6 LSB.

Placing an external capacitor on each analog input also reduces the sensitivity to noise because the capacitor combines with series resistance in the external circuit to form a low-pass filter. In practice, one should include a small series resistance prior to the external capacitor on the analog input pin and choose the largest capacitor value practical, given the frequency of the signal being converted. This provides a low-pass filter on the input, while the resistor also limits input current during overvoltage conditions.

11.6.4 Analog References

To achieve maximum noise isolation, on-chip A/D converters typically separate the internal A/D power supply from the rest of the microcontroller's power supply lines. Separate supply

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

pins, V_{ref} and An_{gnd}, usually supply both the reference and digital voltages for the A/D converter. Keep in mind that V_{ref} and An_{gnd} are the reference for a large resistor ladder on successive approximation converters. Any variation in these supplies will directly affect the reference voltage taps within the ladder, which in turn directly affect A/D conversion accuracy.

If the on-chip A/D converter is not being used, or if accuracy is not a concern, the Vref and Angnd pins can simply be connected to V_{cc} and V_{ss}, respectively. However, since the reference supply levels strongly influence the absolute accuracy of the A/D converter, a precision, well-regulated reference should be used to supply V_{ref} to achieve the highest performance levels. It is also important to use bypass capacitors between V_{ref} and An_{gnd} to minimize any noise that may be present on these supplies. In noise-sensitive applications running at higher frequencies, the use of separate ground planes within the PCB (circuit board) should be considered, possibly as shown in Fig. 11.48. This will help minimize ground loops and provide for a stable A/D reference.

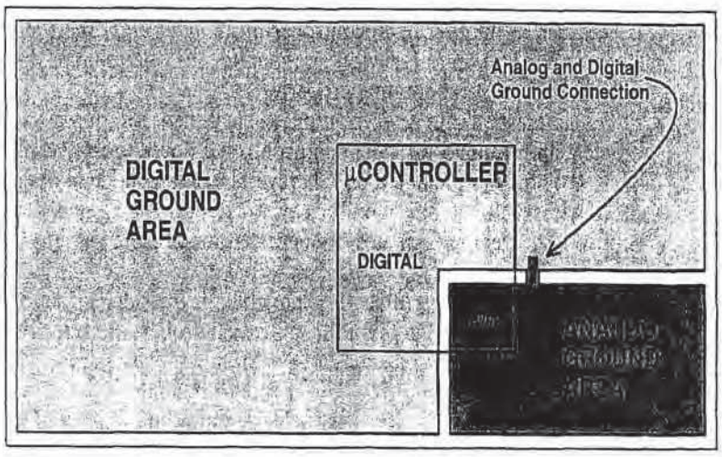


FIGURE 11.48 Example of separate analog and digital ground planes.

11.7 FAILSAFE METHODOLOGIES

The amount and complexity of automotive electronics incorporated into automobiles has increased at an incredible rate over the last decade. This trend has contributed significantly towards the impressive safety record of modern automobiles. Although microcontrollers are extremely reliable electronic devices, it is possible for failures to occur, either elsewhere in the module or within the microcontroller itself. It is critical that these failures be detected and responded to as quickly as possible in safety-related applications such as automotive antilock braking. If proper failsafe methodologies and good programming practices are followed, the chances of a failure going undetected are drastically reduced. The application of *failure mode and effect analysis* (FMEA) is an excellent tool for identifying potential failure modes, detection strategies, and containment methods. Used properly, FMEA will assist the designer in providing a high-quality, reliable automotive module. Although the scope of this chapter does not provide for a discussion on this topic, the author highly encourages the use of FMEA.

11.7.1 Hardware Failsafe Methods

Sometimes a hardware solution is required for detection of and response to certain failure modes. It is difficult for software alone to detect failures external to the device. As an example, consider a case in which electrical overstress (EOS) has damaged a port pin, causing it to

ENGR & MATH SCI LIBRARY UNLV

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

read or drive an incorrect value. In this case, it can be difficult for software to detect because it would base its response on an incorrect value read from a pin.

Watchdog Timers (WDTs). An on-chip hardware watchdog is an excellent method of detecting failures which otherwise may go undetected. An example of this would be a microcontroller fetching either erroneous address or data (due to noise, etc.) and becoming "lost." WDTs commonly utilize a dedicated 16-bit counter, which provides for a count of 2^{16}(65,536) clocked at a rate of one tick per state time. If users wish to take advantage of this feature, they simply write to a register to enable the count. Once enabled, the user program must periodically clear the watchdog by writing a specific bit pattern to the Watchdog SFR. Clearing the WDT at least every 4.1 ms (65,535 * 1 state time at 16 MHz) will prevent the device from being reset. The strategy is that if the WDT initiates a reset, the assumption can be made that a failure has occurred and the microcontroller has became lost.

External Failsafe Devices. It is common for systems to incorporate an external failsafe device, such as another microcontroller or an *application-specific integrated circuit* (ASIC). The function of a failsafe device is to monitor the operation of the primary microcontroller and determine if it is operating properly.

The simplest failsafe devices output a signal such as a square wave for the microcontroller to detect and respond to. If the microcontroller doesn't respond correctly, a reset is typically asserted by the failsafe and the ECU reverts to a safe mode of operation. More complex failsafe devices will actually monitor several critical functions for failures such as low Vcc, stopped or decreased oscillator frequency, shorted/opened input signals, and so forth.

Oscillator Failure Detection. It is possible for the clocking source (typically an oscillator) to fail for various reasons. Since most microcontrollers are static devices, a particularly difficult failure mode to detect is the clocking of the device at a reduced frequency. To detect this failure, an *oscillator failure detection* circuit is often integrated upon the microcontroller. This circuitry will detect if the oscillator clock input signal falls below a specified frequency, in which case an interrupt will be generated or the device will reset itself.

Redundancy/Cross-checking. A common failsafe methodology is achieved by designing a redundant, or backup, processor into the module. In this case, the secondary microcontroller usually executes a subset of the main microcontroller's code. The secondary microcontroller typically processes critical input data and performs cross-checks periodically with the main microcontroller to insure proper operation. A failsafe routine is initiated if data exchanged between the two devices did not correlate.

11.7.2 Software Failsafe Techniques

Failsafe methodologies implemented in software are ideal for detecting failure modes that can interfere with proper program flow. Examples of these types of failures include noise glitches, which are notorious for causing external memory systems to fetch invalid addresses. ROM/EPROM memory corruption could cause an ISR start address to be fetched from an invalid interrupt vector location. Interrupts occurring at a rate faster than anticipated can cause problems such as an overflowing stack. Fortunately, failure modes such as these can be dealt with by implementing software failsafe methods. It is simply good programming practice to anticipate these types of failure modes and provide a failsafe strategy to deal with them. Following are several software strategies commonly used to deal with specific types of failure modes:

Checksum. One possible error that must be accounted for is ROM/EPROM memory corruption. An effective method of detecting these types of failures is through the calculation of

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

a checksum during the initialization phase of a user's program. A checksum is the final value obtained as the result of performing some arithmetic operation upon every ROM/EPROM memory location. The obtained checksum is then compared against a stored checksum. If the two match, the ROM/EPROM contents are intact. An error routine is called if the two checksums do not match. The most common arithmetic operation used to perform a checksum is addition. The checksum is calculated by adding the contents of all memory locations. When the addition is performed, the carry is ignored which provides for a byte or word checksum. The final result is then used as the checksum.

Unused Interrupt Vectors. It is a rare occasion when all interrupt sources are enabled within an application. If, for some unforeseen reason, the program should vector to an unused interrupt source, some sort of failsafe routine should be implemented to respond to the failure. The failsafe routine could be as simple as vectoring to a reset instruction or it can be as complicated as the programmer wishes.

Unused Memory Locations. A strategy should be in place to detect if, for some unforeseen reason, the program sequence should begin to execute in an unused area of ROM/EPROM. It is uncommon for the user's code to fill the entire ROM/EPROM array of a microcontroller. It is good programming practice to fill any unused locations with the opcode of an instruction such as *Reset.* On the MCS-96 family, executing the opcode FFh (which happens to be the blank state of EPROM) will initiate a reset sequence. Other microcontroller families have similar instructions.

Unimplemented Opcode Interrupt Vectors. Microcontrollers often dedicate one or more interrupt vectors for failsafe purposes. An *unimplemented opcode* interrupt is designed to detect corrupted instruction fetches. The corresponding interrupt service routine is executed whenever an unsupported opcode is fetched for execution. The interrupt service routine contains the user's failsafe routine, which is tailored to address this failure for the specific application.

11.8 FUTURE TRENDS

There are several significant trends developing in automotive electronics as ECU manufacturers strive to meet the challenges of a demanding automotive electronics market. The challenges that are bringing about these trends are: decreasing cost targets, decreasing form-factor goals, increasing performance requirements, and increasing system-to-system communication requirements. As the most significant component of an ECU, microcontrollers are bearing the brunt of these demands. This section will discuss these challenges and provide some insight into some of the ways microcontroller manufacturers are addressing these trends.

11.8.1 Decreasing Cost Targets

Microcontroller manufacturers are aproaching cost reduction in two ways: indirectly and directly. *Indirect* cost reductions are achieved by integrating features onto the microcontroller which allow the system designer to reduce cost elsewhere in the system. The key to this approach being successful is in the microcontroller manufacturer's ability to integrate the feature cheaper than the cost of providing an external solution. Integration is not always the cheaper solution, therefore each feature must be evaluated individually to determine the feasability of integration. An example of an indirect cost reduction would be the integration of watchdog and failsafe functions onto the microcontroller. This would eliminate the need for external watchdog components and thus reduce cost.

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Another example would be through the integration of communications protocols such as CAN (Controller Area Network) or J1850 onto the same piece of silicon as the microcontroller. This will reduce the system chip count (and thus cost) by at least one integrated circuit device (the CAN chip) and several interfacing components. In most cases, a reduced chip count will translate into a PCB size decrease and a cost savings.

By *directly* addressing decreasing cost targets, microcontroller manufacturers actually reduce the manufacturing cost of the microcontroller itself. An example of this would be utilizing smaller geometry processes for manufacturing. Process geometry refers to the transistor channel width that is implanted onto a piece of silicon for a given fabrication process. Smaller processes allow for a higher transistor density on an integrated circuit. Higher densities allow for smaller die sizes which relate to lower costs. Most automotive microcontrollers manufactured today are fabricated with a 1.0-micron, or larger, process. As technology advances, future automotive microcontrollers will be manufactured upon submicron processes, such as 0.6 micron.

11.8.2 Increasing Performance Requirements

Automotive applications, such as ABS and engine control, require the processing of a substantial amount of data within a limited period of time. Higher-performance microcontrollers are required as system complexity increases and new features, such as traction control and vehicle dynamics, are incorporated into the ECU.

Microcontroller performance can be directly related to speed. Therefore, a rather straightforward approach to increased performance is through increasing clock speed. Today, most automotive microcontrollers have the capability to operate at frequencies of 16 MHz with speeds up to 20 MHz becoming common. Future microcontrollers will have the ability to be operated at frequencies of 24 or even 32 MHz. This allows more code to be executed in the same amount of time, and thus improves performance.

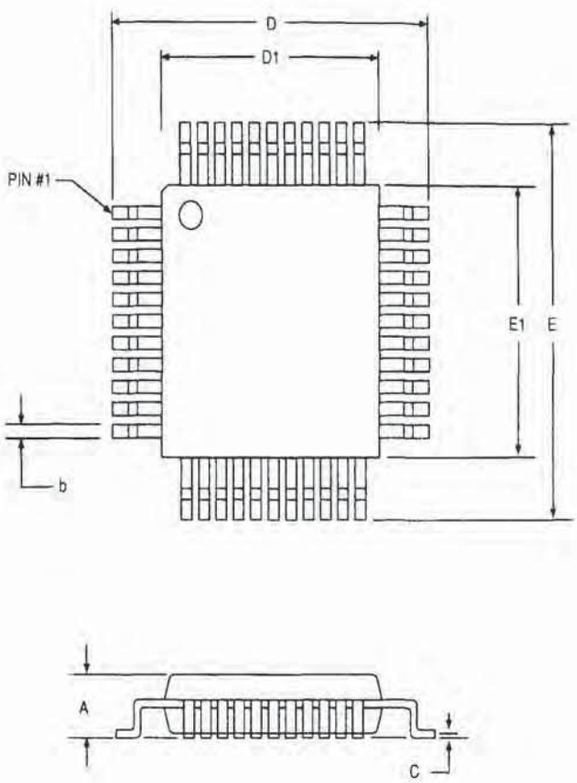
The method of increasing performance is not limited to just increasing the clock frequency. Microcontrollers can also achieve higher performance by enhancing existing peripherals for more efficient operation. This may be in the form of improved data handling or new features which suit the needs of a specific automotive application.

11.8.3 Increasing System-to-System Communication Requirements

The increasing complexity of automotive electronics requires that an increasing amount of information (diagnostics, etc.) be shared between various ECUs within an automobile. To fulfill this need, high-speed data links are utilized to transfer messages between multiple ECUs utilizing protocols such as Bosch's Controller Area Network (CAN) and SAE's J1850. To provide further size and cost savings, it is becoming more and more common to see these protocols supported or integrated onto automotive microcontrollers as opposed to separate integrated circuits.

The theory of centralized body computing is also receiving a closer look due to increased government regulations concerning fuel economy and diagnostics. A centralized body computer would link all ECUs (ABS and traction control, engine, transmission, suspension, instrumentation, etc.) together over a high-speed, in-vehicle serial network. One common scenario would have the central computer (possibly a microprocessor as opposed to a microcontroller) performing the more intense data-crunching tasks, while peripheral microcontrollers located in each individual ECU would perform system I/O functions. These communication protocols provide for efficient two-wire, high-speed serial communications between multiple ECUs utilizing protocols such as CAN and J1850. Supporting these protocols places additional loading upon the microcontroller. Increased microcontroller performance is necessary to manage this loading.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540



SHRINK QUAD FLATPACK				
SYMBOL	DESCRIPTION	MIN.	NOM.	MAX.
N	Lead Count		80	
A	Overall Height			1.66
A1	Stand Off	0.00		
b	Lead Width	0.14	0.20	0.26
c	Lead Thickness	0.117	0.127	0.177
D	Terminal Dimension	13.70	14.00	14.30
D1	Package Body		12.0	
E	Terminal Dimension	13.70	14.00	14.30
E1	Package Body		12.0	
e1	Lead Pitch	0.40	0.50	0.60
L1	Foot Length	0.35	0.50	0.70
T	Lead Angle	0.0°		10.0°
Y	Coplanarity			0.10

FIGURE 11.49 Shrink quad flat pack (SQFP) package.

ENGR & MATH SCI LIBRARY UCLA

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

11.8.4 Decreasing Form Factor Goals

Automobile manufacturers striving to build compact, more fuel efficient automobiles are putting pressure upon ECU suppliers to build smaller, lighter modules.

ECU size is directly affected by PCB size. The easiest way to achieve a smaller PCB is through integration and utilization of smaller integrated circuit packages. To support this demand, automotive microcontroller manufacturers are beginning to offer smaller, fine-pitch packages. A package commonly used today is the 68-lead plastic leaded chip carrier (PLCC) which has its pins placed on 1.27-mm centers and a body that is 24.3 mm^2. An example of a possible automotive package solution for the future would be the 80-lead shrink quad flat pack (SQFP, Fig. 11.49) which has pins on 0.50-mm centers and a body that is 12.0 mm^2. It is relatively easy to see that the SQFP package offers 12 additional pins in a package that is half the size of the PLCC. This high pin density, fine-pitch packaging allows for a smaller package to be utilized for the same size microcontroller die.

Another technology that is quickly becoming popular for automotive applications is referred to as *multichip modules* (MCMs). An MCM is a collection of unpackaged integrated circuit die (from various manufacturers) which are mounted upon a common substrate and packaged together. The advantage of MCMs is that they require much less PCB space than if the ICs were packaged separately.

GLOSSARY

Accumulator A register within a microcontroller that holds data, particularly data on which arithmetic or logic operations are to be performed.

Arithmetic logic unit (ALU) The part of a microcontroller that performs arithmetic and logic operations.

Analog-to-digital converter An electronic device that produces a digital result that is proportional to the analog input voltage.

Assembly language A low-level symbolic programming language closely resembling machine language.

Central processing unit (CPU) The portion of a computer system or microcontroller that controls the interpretation and execution of instructions and includes arithmetic capability.

EPROM Erasable and programmable read-only memory.

High-speed input/output unit (HSIO) A microcontroller peripheral which has the capability to either capture the time at which a certain input event occurs or create an output event at a predetermined time, both relative to a common clock. HSIO events are configured by the programmer to occur automatically.

Interrupt service routine (ISR) A predefined portion of a computer program which is executed in response to a specific event.

Low-speed input/output The input/output of a digital signal by "manually" reading or writing a register location in software.

Machine language A set of symbols, characters, or signs used to communicate with a computer in a form directly usable by the computer without translation.

Program counter (PC) A microcontroller register which holds the address of the next instruction to be executed.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Program status word (PSW) A microcontroller register that contains a set of boolean flags which are used to retain information regarding the state of the user's program.

Pulse-width modulation (PWM) The precise and timely creation of negative and positive waveform edges to achieve a waveform with a specific frequency and duty cycle.

Random access memory (RAM) A memory device which has both read and write capabilities so that the stored information (write) can be retrieved (reread) and be changed by applying new information to the inputs.

Read-only memory (ROM) A memory that can only be read and not written to. Data is either entered during the manufacturing process or by later programming; once entered, it is unalterable.

Register/arithmetic logic unit (RALU) A component of register-direct microcontroller architectures that allows the ALU to operate directly upon the entire register file.

Serial input/output (SIO) A method of digital communication in which a group of data bits is transferred one at a time, sequentially over a single data line.

Special function register (SFR) A microcontroller RAM register which has a specific, dedicated function assigned to it.

BIBLIOGRAPHY

ASM96 Assembler User's Manual, Intel Corp., 1992.

Automotive Electrics/Electronics, Robert Bosch GmbH, 1988.

Automotive Handbook, Intel Corporation, 1994.

Automotive Handbook, 2d ed., Robert Bosch GmbH, 1986.

Corell, Roger J., "How are semiconductor suppliers responding to the growing demand for automotive safety features?," *Intel Corp.,* 1993.

Davidson, Lee S., and Robert M. Kowalczyk, "Microcontroller technology enhancements to meet ever-increasing engine control requirements," Intel Corp., 1992.

Fink, Donald G., and Donald Christiansen, *Electronics Engineers' Handbook,* 3d ed. McGraw-Hill, 1989.

iC-96 Compiler User's Manual, Intel Corp., 1992.

Introduction to MOSFETS and EPROM Memories, Intel Corp., 1990.

MCS®-51 Microcontroller Family User's Manual, Intel Corp., 1993.

Millman, Jacob, and Arvin Grabel, *Microelectronics,* McGraw-Hill, 1987.

Packaging Handbook, Intel Corporation, 1994.

Ribbens, William B., *Understanding Automotive Electronics,* Howard Sams Company, Carmel, Ind. 1992.

8XC196Kx User's Manual, Intel Corporation, 1992.

8XC196KC/8XC196KD User's Manual, Intel Corp., 1992.

ABOUT THE AUTHOR

David S. Boehmer is currently a senior technical marketing engineer for the Automotive Operation of Intel's Embedded Microprocessor Division located in Chandler, Ariz. He is a member of SAE.

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

CHAPTER 12
ENGINE CONTROL

Gary C. Hirschlieb, Gottfried Schiller, and Shari Stottler
Robert Bosch GmbH

ENGR & MATH SCI LIBRARY

12.1 OBJECTIVES OF ELECTRONIC ENGINE CONTROL SYSTEMS

The electronic engine control system consists of sensing devices which continuously measure the operating conditions of the engine, an electronic control unit (ECU) which evaluates the sensor inputs using data tables and calculations and determines the output to the actuating devices, and actuating devices which are commanded by the ECU to perform an action in response to the sensor inputs.

The motive for using an electronic engine control system is to provide the needed accuracy and adaptability in order to minimize exhaust emissions and fuel consumption, provide optimal driveability for all operating conditions, minimize evaporative emissions, and provide system diagnosis when malfunctions occur.

In order for the control system to meet these objectives, considerable development time is required for each engine and vehicle application. A substantial amount of development must occur with an engine installed on an engine dynamometer under controlled conditions. Information gathered is used to develop the ECU data tables. A considerable amount of development effort is also required with the engine installed in the vehicle. Final determination of the data tables occurs during vehicle testing.

12.1.1 Exhaust Emissions

Exhaust Components. The engine exhaust consists of products from the combustion of the air and fuel mixture. Fuel is a mixture of chemical compounds, termed hydrocarbons (HC). The various fuel compounds are a combination of hydrogen and carbon. Under perfect combustion conditions, the hydrocarbons would combine in a thermal reaction with the oxygen in the air to form carbon dioxide (CO_2) and water (H_2O). Unfortunately, perfect combustion does not occur and in addition to CO_2 and H_2O, carbon monoxide (CO), oxides of nitrogen (NO_x), and hydrocarbons (HC) occur in the exhaust as a result of the combustion reaction. Additives and impurities in the fuel also contribute minute quantities of pollutants such as lead oxides, lead halogenides, and sulfur oxides. In compression ignition (diesel) engines, there is also an appreciable amount of soot (particulates) created. Federal statues regulate the allowable amount of HC, NO_x, and CO emitted in a vehicle's exhaust. On diesel engines, the amount of particulates emitted is also regulated.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Spark Ignition Engines

Air/fuel Ratio. The greatest effect on the combustion process, and therefore on the exhaust emissions, is the mass ratio of air to fuel. The air/fuel mixture ratio must lie within a certain range for optimal ignition and combustion. For a spark ignition engine, the mass ratio for complete fuel combustion is 14.7:1; i.e., 14.7 kg of air to 1 kg of fuel. This ratio is known as the stoichiometric ratio. In terms of volume, approximately 10,000 liters of air would be required for 1 liter of fuel. The air/fuel ratio is often described in terms of the excess-air factor known as lambda (λ). Lambda indicates the deviation of the actual air/fuel ratio from the theoretically required ratio:

$$\lambda = \frac{\text{quantity of air supplied}}{\text{theoretical requirement (14.7 for gasoline)}}$$

At stoichiometry: $\lambda = 1$
For a mixture with excess air (lean): $\lambda > 1$
For a mixture with deficient air (rich): $\lambda < 1$

Effect of Air/Fuel Ratio on Emissions

CO emissions. In the rich operating range ($\lambda < 1$), CO emissions increase almost linearly with an increasing amount of fuel. In the lean range ($\lambda > 1$), CO emissions are at their lowest. With an engine operating at ($\lambda = 1$), the CO emissions can be influenced by the cylinder distribution. If some cylinders are operating rich and others lean with the summation achieving $\lambda = 1$, the average CO emissions will be higher than if all cylinders were operating at $\lambda = 1$.

HC emissions. As with CO emissions, HC emissions increase with an increasing amount of fuel. The minimum HC emissions occur at $\lambda = 1.1 \ldots 1.2$. At very lean air/fuel ratios, the HC emissions again increase due to less than optimal combustion conditions resulting in unburned fuel.

NO_x emissions. The effect of the air/fuel ratio on NO_x emissions is the opposite of HC and CO on the rich side of stoichiometry. As the air content increases, the oxygen content increases and the result is more NO_x. On the lean side of stoichiometry, NO_x emissions decrease with increasing air because the decreasing density lowers the combustion chamber temperature. The maximum NO_x emissions occur at $\lambda = 1.05 \ldots 1.1$.

Catalytic Converters. To reduce the exhaust gas emission concentration, a catalytic converter is installed in the exhaust system. Chemical reactions occur in the converter that transform the exhaust emissions to less harmful chemical compounds. The most commonly used converter for a spark ignition engine is the three-way converter (TWC). As the name implies, it simultaneously reduces the concentration of all three regulated exhaust gases: HC, CO, and NO_x. The catalyst promotes reactions that oxidize HC and CO, converting them into CO_2 and H_2O, while reducing NO_x emissions into N_2. The actual chemical reactions that occur are:

$$2CO + O_2 \rightarrow 2CO_2$$

$$2C_2H_6 + 7O_2 \rightarrow 4CO_2 + 6H_20$$

$$2NO + 2CO \rightarrow N_2 + 2CO_2$$

In order for the catalytic converter to operate at the highest efficiency for conversion for all three gases (HC, CO, NO_x), the average air/fuel ratio must be maintained within less than 1 percent of stoichiometry. This small operating range is known as the *lambda window* or *catalytic converter window.* Figure 12.1 is a graph of lambda (λ) versus the exhaust emissions both before and after the catalytic converter. Up to 90 percent of the exhaust gases are converted to less harmful compounds by the catalytic converter.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

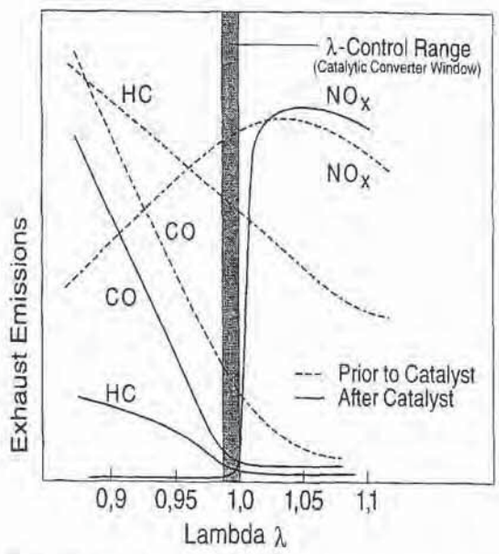


FIGURE 12.1 Lambda effect on exhaust emissions prior to and after catalyst treatment.

To remain within the catalytic converter window, the air/fuel ratio is controlled by the lambda closed-loop fuel control system, which is part of the electronic engine control system. The key component in this system is the lambda sensor. This sensor is installed in the exhaust system upstream of the catalytic converter and responds to the oxygen content in the exhaust gas. The oxygen content is a measure of the excess air (or deficiency of air) in the exhaust gases. A detailed discussion of the lambda closed-loop control system occurs in Sec. 12.2.1.

Ignition Timing. The ignition timing is defined as the crankshaft angle before top dead center (TDC) at which the ignition spark occurs. The ignition timing of the air/fuel mixture has a decisive influence on the exhaust emissions.

Effect of ignition timing on exhaust emissions.

- CO emissions are almost completely independent of the ignition timing and are primarily a function of the air/fuel ratio.
- In general, the more the ignition is advanced, the higher the emissions of HCs. Reactions initiated in the combustion chamber continue to occur after the exhaust valve opens, which depletes the remaining HCs. With advanced timing due to lower exhaust temperatures, these postreactions do not readily occur.
- With increased timing advance, the combustion chamber temperatures increase. The temperature increase causes an increase in NO_x emissions regardless of air/fuel ratio.

To provide the optimal ignition timing for exhaust emissions, precise control of the ignition timing is required. It is imperative that the ignition timing be coordinated with the air/fuel ratio since they have a combined effect on exhaust emissions as well as fuel consumption and driveability. Ignition timing is generally controlled by the ECU. Ignition timing control is discussed in detail in Sec. 12.2.1.

Exhaust Gas Recirculation (EGR). Exhaust gas recirculation (EGR) is a method of reducing emissions of oxides of nitrogen. A portion of the exhaust gas is recirculated back to the combustion chamber. Exhaust gas is an inert gas and, in the combustion chamber, it lowers the peak combustion temperature. Depending on the amount of EGR, NO_x emissions can be reduced by up to 60 percent, although an increase in HC emissions would occur at such high levels of EGR.

Some internal EGR occurs due to the overlap of the exhaust and intake valves. Additional quantities are supplied by a separate system linking the exhaust manifold to the intake mani-

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

fold. The quantity of EGR flow to the intake system is metered by a pneumatic or electronic valve. The EGR valve is controlled by the ECU. The maximum flow of EGR is limited by an increase in HC emissions, fuel consumption, and engine roughness. EGR control is discussed in detail in Sec. 12.2.1.

Compression Ignition (Diesel) Engines. There are some key distinctions between an SI engine and a CI engine. The CI engine uses high pressure and temperature instead of a spark to ignite the combustible air/fuel mixture. To achieve this, the CI engine compression ratio is in the range of 21:1, as opposed to roughly 10:1 for an SI engine. In a CI engine, the fuel is injected directly into the cylinder near the top of the compression stroke. Mixing of the fuel and air, therefore, occurs directly in the cylinder.

Air/fuel ratio. Diesel engines always operate with excess air ($\lambda > 1$). Where:

$$\lambda = \frac{\text{quantity of air supplied}}{\text{theoretical requirement}}$$

The excess air ($\lambda = 1.1 \ldots 1.2$) reduces the amount of soot (particulates), HC, and CO emissions.

Catalytic Converters. An oxidizing catalyst is used that converts CO and HC to CO_2 and H_2O. The NO_x reduction that occurs for an SI engine three-way catalyst (TWC) is not possible with a diesel because the diesel operates with excess air. The optimal conversion of NO_x requires a stoichiometric ratio ($\lambda = 1$) or a deficiency of air ($\lambda < 1$).

Injection Timing. In a compression ignition engine, the start of combustion is determined by the start of fuel injection. In general, retarding the injection timing decreases NO_x emissions, while overretarding results in an increase in HC emissions. A 1° (crankshaft angle) deviation in injection timing can increase NO_x emissions by 5 percent and HC emissions by as much as 15 percent. Precise control of injection timing is critical. Injection timing on some systems is controlled by the ECU. Feedback on injection timing can be provided by a sensor installed on the injector nozzle. Further discussion on injection timing occurs in Sec. 12.3.1.

Exhaust Gas Recirculation (EGR). As with an SI engine, exhaust gas can be recirculated to the combustion chamber to significantly reduce NO_x emissions. The quantity of EGR allowed to enter the intake is metered by the EGR valve. If the quantity is too high, HC emissions, CO emissions, and soot (particulates) increase as a result of an insufficient quantity of air. The EGR valve is controlled by the ECU, which determines how much EGR is tolerable under the current engine operating conditions.

12.1.2 Fuel Consumption

Federal statutes are currently in effect that require each automobile manufacturer to achieve a certain average fuel economy for all their models produced in one model year. The requirement is known as *corporate average fuel economy* or CAFE. The fuel economy for each vehicle type is determined during the federal test procedure, the same as for exhaust emissions determination, conducted on a chassis dynamometer. Because of the CAFE requirement, it is critical that fuel consumption be minimized for every vehicle type produced.

The electronic engine control system provides the fuel metering and ignition timing precision required to minimize fuel consumption. Optimum fuel economy occurs near $\lambda = 1.1$. However, as discussed previously, lean engine operation affects exhaust emissions and NO_x is at its maximum at $\lambda = 1.1$.

During coasting and braking, fuel consumption can be further reduced by shutting off the fuel until the engine speed decreases to slightly higher than the set idle speed. The ECU determines when fuel shutoff can occur by evaluating the throttle position, engine RPM, and vehicle speed.

The influence of ignition timing on fuel consumption is the opposite of its influence on exhaust emissions. As the air/fuel mixture becomes leaner, the ignition timing must be advanced to compensate for a slower combustion speed. However, as discussed previously,

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

advancing the ignition timing increases the emissions of HC and NO_x. A sophisticated ignition control strategy permitting optimization of the ignition at each operating point is necessary to reach the compromise between fuel consumption and exhaust emissions. The electronic engine control system can provide this sophisticated strategy.

12.1.3 Driveability

Another requirement of the electronic engine control system is to provide acceptable driveability under all operating conditions. No stalls, hesitations, or other objectionable roughness should occur during vehicle operation. Driveability is influenced by almost every operation of the engine control system and, unlike exhaust emissions or fuel economy, is not easily measured. A significant contribution to driveability is determined by the fuel metering and ignition timing. When determining the best fuel and ignition compromises for fuel consumption and exhaust emissions, it is important to evaluate the driveability. Other factors that influence driveability are the idle speed control, EGR control, and evaporative emissions control.

12.1.4 Evaporative Emissions

Hydrocarbon (HC) emissions in the form of fuel vapors escaping from the vehicle are closely regulated by federal statutes. The prime source of these emissions is the fuel tank. Due to ambient heating of the fuel and the return of unused hot fuel from the engine, fuel vapor is generated in the tank. The evaporative emissions control system (EECS) is used to control the evaporative HC emissions. The fuel vapors are routed to the intake manifold via the EECS and they are burned in the combustion process. The quantity of fuel vapors delivered to the intake manifold must be metered such that exhaust emissions and driveability are not adversely affected. The metering is provided by a purge control valve whose function is controlled by the ECU. Further discussion on the operation of the evaporative emissions control system occurs in Sec. 12.2.1.

12.1.5 System Diagnostics

The purpose of system diagnostics is to provide a warning to the driver when the control system determines a malfunction of a component or system and to assist the service technician in identifying and correcting the failure (see Chap. 22). To the driver, the engine may appear to be operating correctly, but excessive amounts of pollutants may be emitted. The ECU determines a malfunction has occurred when a sensor signal received during normal engine operation or during a system test indicates there is a problem. For critical operations such as fuel metering and ignition control, if a required sensor input is faulty, a substitute value may be used by the ECU so that the engine will continue to operate.

When a failure occurs, the malfunction indicator light (MIL), visible to the driver, is illuminated. Information on the failure is stored in the ECU. A service technician can retrieve the information on the failure from the ECU and correct the problem. Detailed examples of system diagnostics are discussed in Sec. 12.2.3.

12.2 SPARK IGNITION ENGINES

12.2.1 Engine Control Functions

Fuel Control. For the purpose of discussing fuel control strategies, a multipoint pulsed fuel injection system is assumed. Additional discussions of fuel control for different types of fuel

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

systems such as carbureters, single-point injection, and multipoint continuous injection appear in Sec. 12.2.4 (Fuel Delivery Systems).

In order for the fuel metering system to provide the appropriate amount of fuel for the engine operating conditions, the mass flow rate of incoming air, known as the air charge, must be determined.

$$F_m = \frac{A_m}{\text{requested air-fuel ratio}}$$

where F_m = fuel mass flow rate
A_m = air mass flow rate

The air mass flow rate can be calculated from:

$$A_m = A_v A_d$$

where A_v = volume flow rate of intake air
A_d = air density

There are three methods commonly used for determining the air charge: speed density, air flow measurement, and air mass measurement. In the speed density method, the air charge is calculated by the engine electronic control unit based on the measurement of air inlet temperature, intake manifold pressure, and engine RPM. The temperature and pressure are used to determine the air density and the RPM is used to determine the volume flow rate. The engine acts as an air pump during the intake stroke. The calculated volume flow rate can be determined as follows:

$$A_{\text{RPM}} = \frac{\text{RPM}}{60} \times \frac{D}{2} \times V_{\text{E}}$$

where RPM = engine speed
D = engine displacement
V_{E} = volumetric efficiency

In an engine using exhaust gas recirculation (EGR), the volume flow rate of EGR must be subtracted from the calculated volume flow rate.

$$A_v = A_{\text{RPM}} - A_{\text{EGR}}$$

The volume flow rate of EGR can be determined empirically based on the EGR valve flow rate and the EGR control strategy being used.

In the air flow measurement method, the air flow is measured using a vane type meter and air density changes are compensated for by an air inlet temperature sensor. The vane meter uses the force of the incoming air to move a flap through a defined angle. This angular movement is converted by a potentiometer to a voltage ratio. Because only the fresh air charge is measured, no compensation is required for EGR.

In the air mass measurement method, the air charge is measured directly using a hot-wire or hot-film air mass flow sensor. The inlet air passes a heated element, either wire or film. The element is part of a bridge circuit that keeps the element at a constant temperature above the inlet air temperature. By measuring the heating current required by the bridge circuit and converting this to a voltage via a resistor, the air mass flow passing the element can be determined. Again, because only the fresh air charge is measured, no compensation for EGR is required. However, sensing errors may occur due to strong intake manifold reversion pulses, which occur under certain operating conditions. In such cases, a correction factor must be determined and applied.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Calculation of Injector Pulse Width. The base pulse width is determined from the required fuel mass flow rate (F_m) and an empirical injector constant. The injector constant is determined by the design of the injector and is a function of the energized time versus the flow volume. This constant is normally determined with a constant differential pressure across the injector (from fuel rail to intake manifold). When the pressure across the injector does not remain constant (i.e., there is no pressure regulator intake manifold vacuum reference), an entire map of injector constants for different manifold pressures may be required.

The effective injector pulse width is a modification of the base pulse width. The base pulse width is adjusted by a number of correction factors depending on operating conditions. For example, a battery voltage correction is required to compensate for the electromechanical characteristics of the fuel injectors. Injector opening and closing rates differ depending on the voltage applied to the injector, which affects the amount of fuel injected for a given pulse width. Other common correction factors may include hot restart, cold operation, and transient operation corrections. Figure 12.2 is a flowchart of a typical injector effective pulse-width calculation method.

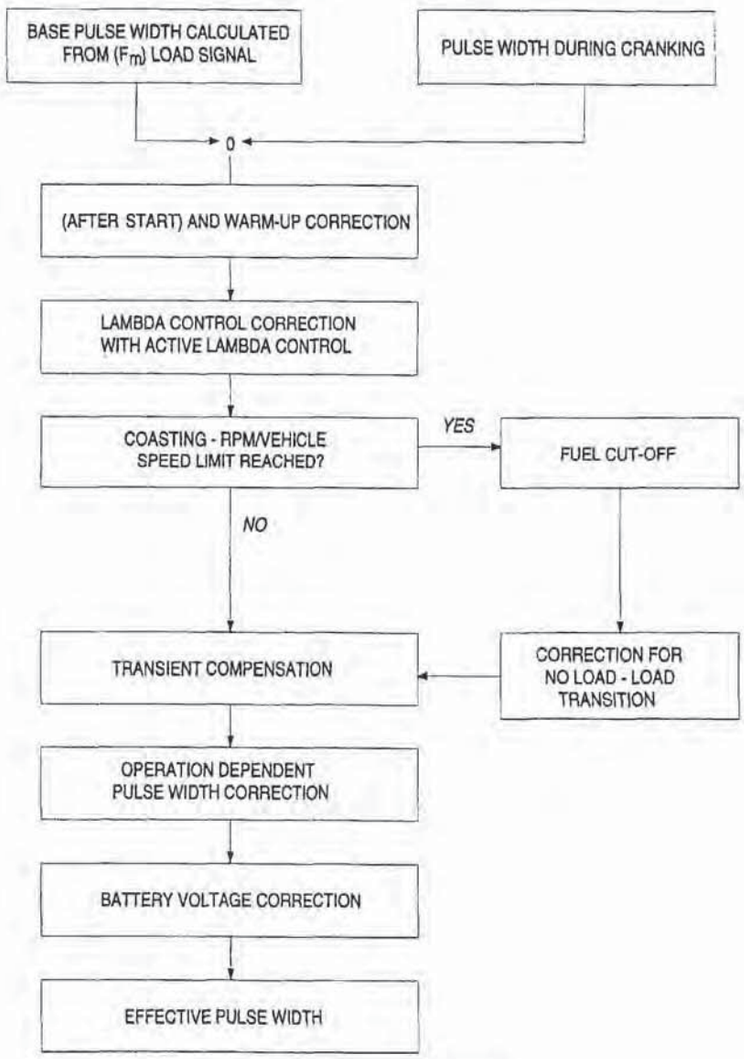


FIGURE 12.2 Determination of effective injector pulse width.

ENGR & MATH SCI LIBRARY UNLV

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Injection Strategies. There are three commonly used fuel injection strategies for multipoint fuel metering systems: simultaneous injection, group injection, and sequential injection. Figure 12.3 is a diagram of the different strategies. Some engines use simultaneous injection during crank and switch over to sequential after the engine is running. This allows for shorter starting times since no synchronization with the camshaft is necessary before fuel injection begins. A description of each strategy follows.

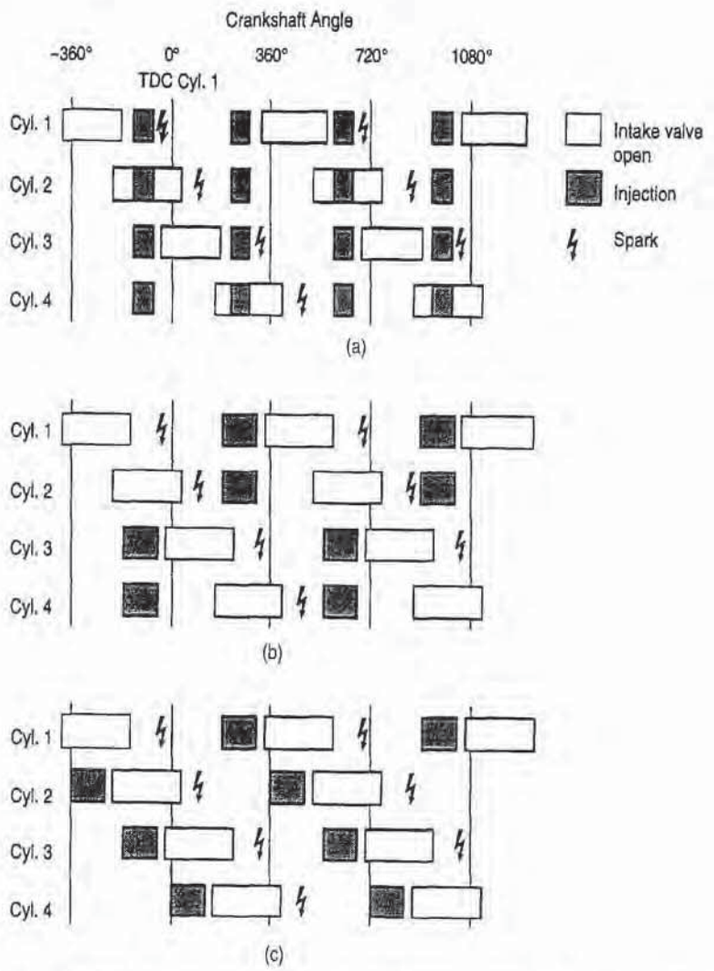


FIGURE 12.3 Fuel injection strategies: (*a*) simultaneous injection, (*b*) group injection, and (*c*) sequential injection.

Simultaneous injection. Injection of fuel occurs at the same time for all cylinders every revolution of the crankshaft. Therefore, fuel is injected twice within each four-stroke cycle. The injection timing is fixed with respect to crank/camshaft position.

Group injection. The injectors are divided into two groups that are controlled separately. Each group injects once per four-stroke cycle. The offset between the groups is one crankshaft revolution. This arrangement allows for injection timing selection that eliminates spraying fuel into an open intake valve.

Sequential injection. Each injector is controlled separately. Injection timing, both with reference to crank/camshaft position and pulse width, can be optimized for each individual cylinder.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Lambda Control. A subsystem of the fuel control system is lambda closed-loop control. Lambda (λ) is defined as the excess-air factor that indicates the deviation of the actual air/fuel ratio from the theoretically required ratio:

$$\lambda = \frac{\text{quantity of air supplied}}{\text{theoretical requirement (14.7 for gasoline)}}$$

The lambda sensor, or exhaust gas oxygen sensor, is installed in the engine exhaust stream upstream of the catalytic converter. The sensor responds to the oxygen content of the exhaust gas. The signal from the lambda sensor serves as feedback to the fuel control system. This provides the fine-tuning needed to remain within the limited catalytic converter window for optimal catalyst performance. (See Sec. 12.1.1 for more discussion on the catalytic converter window.) For a lean mixture ($\lambda > 1$), sensor voltage is approximately 100 mV. For a rich mixture ($\lambda < 1$), the sensor voltage is approximately 800 mV. At roughly $\lambda = 1$ (a stoichiometric mixture), the sensor switches rapidly between the two voltages. The input from the lambda sensor is used to modify the base pulse width to achieve $\lambda = 1$.

Lambda closed-loop control requires an operationally ready lambda sensor, typically one which has reached an operating temperature threshold. Sensor output is monitored by the ECU to determine when the sensor is supplying usable information. An active sensor signal, along with other requirements, such as engine temperature, must be achieved before lambda closed-loop control will be activated.

Under steady state conditions, the lambda control system oscillates between rich and lean around the lambda window. As the lambda sensor switches, the injector pulse width is adjusted by the amount determined by a control factor until the lambda sensor switches again to the opposite condition. The control factor can be defined as the allowable increase or decrease in the commanded fuel injector pulse width. The frequency of oscillation is determined by the gas transport time and the magnitude of the control factor. The gas transport time is defined as the time from air/fuel mixture formation to lambda sensor measurement.

Under transient conditions, the gas transport time results in a delay before the lambda sensor can indicate that the operating conditions have changed. Using only the lambda sensor for closed-loop fuel control would result in poor driveability and exhaust emissions because of this delay. Therefore, the engine control unit uses an anticipatory control strategy that uses engine load and RPM to determine the approximate fuel requirement. The engine load information is provided by the manifold pressure sensor for speed density systems and by the air meter for air flow and air mass measurement systems and by the throttle valve position sensor. The engine control unit contains data tables for combinations of load and RPM. This allows for rapid response to changes in operating conditions. The lambda sensor still provides the feedback correction for each load/RPM point. The data used for these data tables are largely developed from system modeling and engine development testing.

Due to production variations in engines, variations in fuel and changes due to wear and aging, the control system must be able to adapt to function properly for every engine over the engine's life. Therefore, the electronic control unit has a feature for adapting changes in the fuel required for the load/RPM points. At each load/RPM point, the lambda sensor continuously provides information that allows the system to adjust the fuel to the commanded A/F ratio. The corrected information is stored in RAM (random access memory) so that the next time the engine reaches that operating point (load/RPM), the anticipatory value will require less correction. These values remain stored in the electronic control unit even after the engine is shut off. Only if power to the electronic control unit is disrupted (i.e., due to a dead battery), will the correction be lost. In that case, the electronic control unit will revert back to the original production values that are written in ROM (read-only memory).

Lambda sensors do not switch symmetrically from lean to rich and rich to lean. Because of this, the control strategy is modified to account for the asymmetry. This can be accomplished either by delaying the modification by the control factor after the sensor switches or by using control factors of different magnitudes for rich-to-lean and lean-to-rich switching.

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Ignition Timing Control. The goal of the engine control system for ignition timing is to provide spark advance which optimizes engine torque, exhaust emissions, fuel economy, and driveability, and which minimizes engine knock. Data tables with the base ignition timing, depending on engine load and RPM, are stored in ROM in the electronic control unit. The values in these tables are optimized for fuel economy, exhaust emissions, and engine torque. They are developed through engine experimentation, usually with an engine dynamometer. Corrections to the base timing values are needed for temperature effects, EGR, hot restart, barometric pressure, and engine knock. In addition, some systems use ignition timing to vary the engine torque for improvement in automatic transmission shift quality or for idle speed control. Figure 12.4 is a flowchart of a typical ignition timing calculation method.

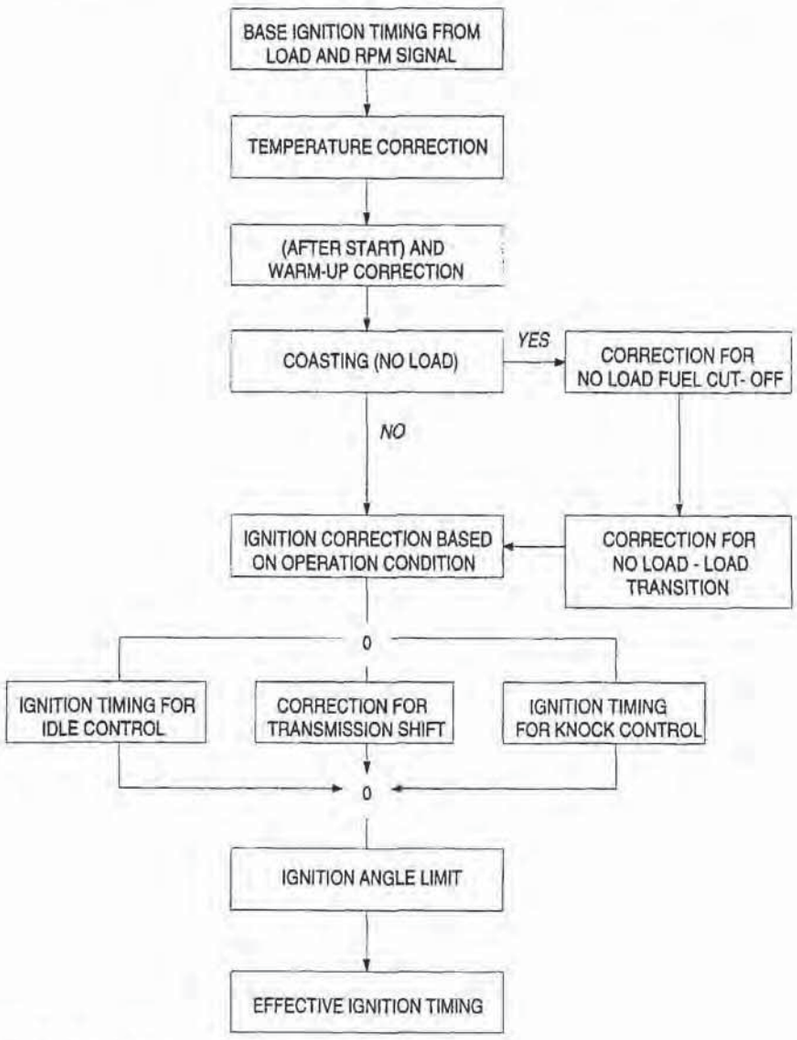


FIGURE 12.4 Determination of effective ignition timing.

Dwell Angle Control. The dwell angle performance map stored in the electronic control unit controls the charging time of the ignition coil, depending on RPM and battery voltage. The dwell angle is controlled so that the desired primary current is reached at the end of the charging time just prior to the ignition point. This assures the necessary primary current, even with quick transients in RPM. A limit on the charge time in the upper RPM ranges allows for the necessary spark duration.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Knock Control. The ignition timing for optimization of torque, fuel economy, and exhaust emissions is in close proximity to the ignition timing that results in engine knock. Engine knock occurs when the ignition timing is advanced too far for the engine operating conditions and causes uncontrolled combustion that can lead to engine damage, depending on the severity and frequency. If a factor of safety was used when developing the base timing map for all conditions that contribute to knock, such as fuel quality and variations in compression ratio, the ignition timing would be significantly retarded from the optimum level, resulting in a significant loss in torque and fuel economy. To avoid this, a knock sensor (one or more) is installed on the engine to detect knocking (see Chap. 8). Knock sensors are usually acceleration sensors that provide an electric signal to the electronic control unit. From this signal, the engine control unit algorithm determines which cylinder or cylinders are knocking. Ignition timing is modified (retarded) for those cylinders until the knock is no longer detected. The ignition timing is then advanced again until knock is detected. (See Fig. 12.5.) Information on the amount of spark retard required to eliminate the knock for each cylinder under each load/RPM condition is saved in the electronic control unit RAM. This allows for quick access to the appropriate "learned" ignition timing for each condition. With this control system, the base timing can be more advanced for improved fuel economy and torque.

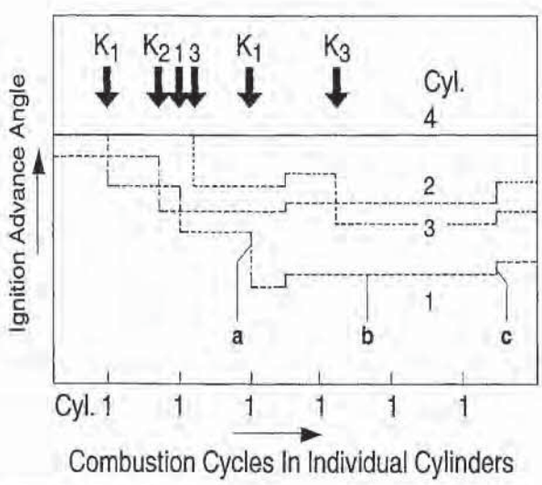


FIGURE 12.5 Knock control. Control algorithm for ignition adjustments for a four-cylinder engine. $K_{1\ldots3}$ (knock in cylinders 1 . . . 3), cylinder number four (no knock), (*a*) (ignition retard), (*b*) (delay before return to original point, (*c*) (spark advance).

Evaporative Emissions Control. Hydrocarbon (HC) emissions in the form of fuel vapors escaping from the vehicle, primarily from the fuel tank, are closely regulated by federal statutes. There are two principal causes of fuel vapor in the fuel tank: increasing ambient temperature and return of unused hot fuel from the engine. In order to control the release of these emissions to the atmosphere, the evaporative emissions control system was developed.

Evaporative Emissions Control System. A vapor ventilation line exits the fuel tank and enters the fuel vapor canister. The canister consists of an active charcoal element which absorbs the vapor and allows only air to escape to the atmosphere. Only a certain volume of fuel vapor can be contained by the canister. The vapors in the canister must therefore be purged from and burned by the engine so that the canister can continue to store vapors as they are generated. To accomplish this, another line leads from the charcoal canister to the intake manifold. Included in this line is the canister purge solenoid valve. Figure 12.6 shows a layout of a typical evaporative emissions control system.

During engine operation, vacuum in the intake manifold causes flow through the charcoal canister because the canister vent opening, at the charcoal filter end, is at atmospheric pres-

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

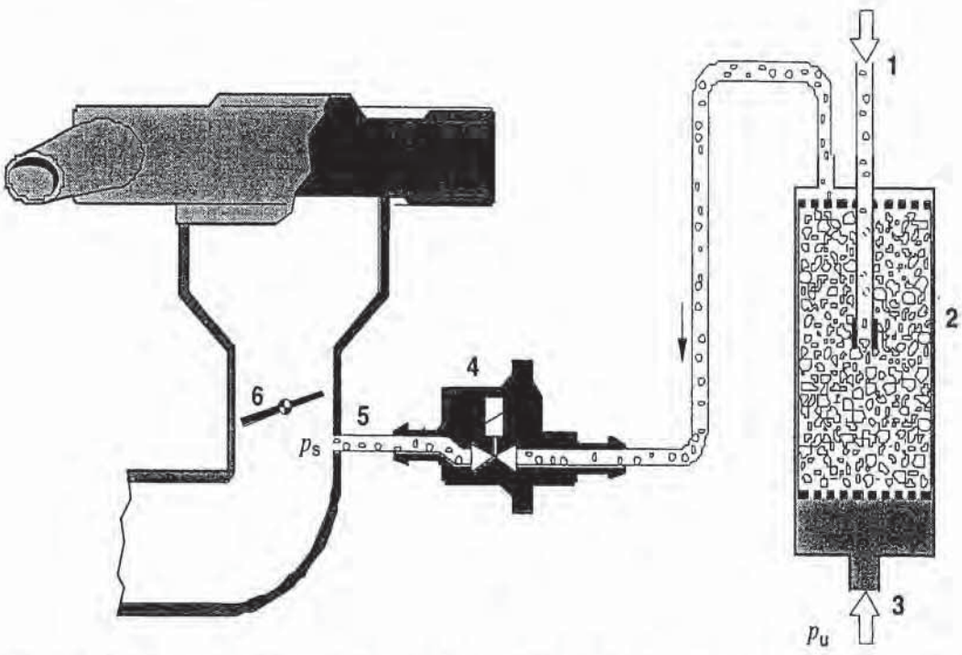


FIGURE 12.6 Evaporative emission control system: fuel vapor from fuel tank (1), charcoal canister (2), ambient air (3), canister purge control valve (4), purge line to intake manifold (5), throttle valve (6), p_s is intake manifold vacuum, and p_u is atmospheric pressure.

sure. The canister purge valve meters the amount of flow from the canister. The amount of fuel vapor in the canister and, therefore, contained in the flow stream, is not known. Therefore, it is critical that the lambda control system is operating and adjusting the fuel requirement as the vapors are being purged. Purge vapors could otherwise result in up to a 30 percent increase in air/fuel mixture richness in the engine.

Purge Valve Control. Control of the purge valve must allow for two criteria:

- There must be enough vapor flow so that the charcoal canister does not become saturated and leak fuel vapors to the atmosphere
- Purge flow must generally occur under lambda closed-loop control so that the effect of the purge vapors on A/F ratio can be detected and the fuel metering corrected

When the electronic control unit commands the purge valve to meter vapor from the canister, it requests a duty cycle (ratio of ON time to total ON and OFF time). This allows the amount of vapor flow to be regulated depending on the engine operating conditions. When lambda control is not operating, only low duty cycles and, therefore, small amounts of purge vapors, are allowed. Under deceleration fuel cutoff, the purge valve is closed entirely to minimize the possibility of unburned HCs in the exhaust.

Turbocharger Boost Pressure Control. The exhaust turbocharger consists of a compressor and an exhaust turbine arranged on a common shaft. Energy from the exhaust gas is converted to rotational energy by the exhaust turbine, which then drives the compressor. The compressed air leaves the compressor and passes through the air cooler (optional), throttle valve, intake manifold, and into the cylinders. In order to achieve near-constant air charge pressure over a wide RPM range, the turbocharger uses a circuit that allows for the bypass of the exhaust gases away from the exhaust turbine. The valve that regulates the bypass opens at a specified air charge pressure and is known as the wastegate.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Engines that have turbochargers benefit significantly from electronic boost pressure control. If only a pneumatic-mechanical wastegate is used, only one boost pressure point for the entire operating range is used to divert the exhaust gas. This creates a compromise for part-load conditions, which results in increased exhaust backpressure, more turbocharger work, more residual exhaust gas in the cylinders, and higher-charge air temperatures.

By controlling the wastegate with a pulse-width modulated solenoid valve, different wastegate opening pressures can be specified, depending on the engine operating conditions (Fig. 12.7). Therefore, only the level of air charge pressure required is developed. The electronic control unit uses information on engine load from either manifold pressure or the air meter and RPM and throttle position. From this information, a data table is referenced and the proper boost pressure (actually a duty cycle of the control valve) is determined. On systems using manifold pressure sensors, a closed-loop control system can be developed to compare the specified value with the measured value.

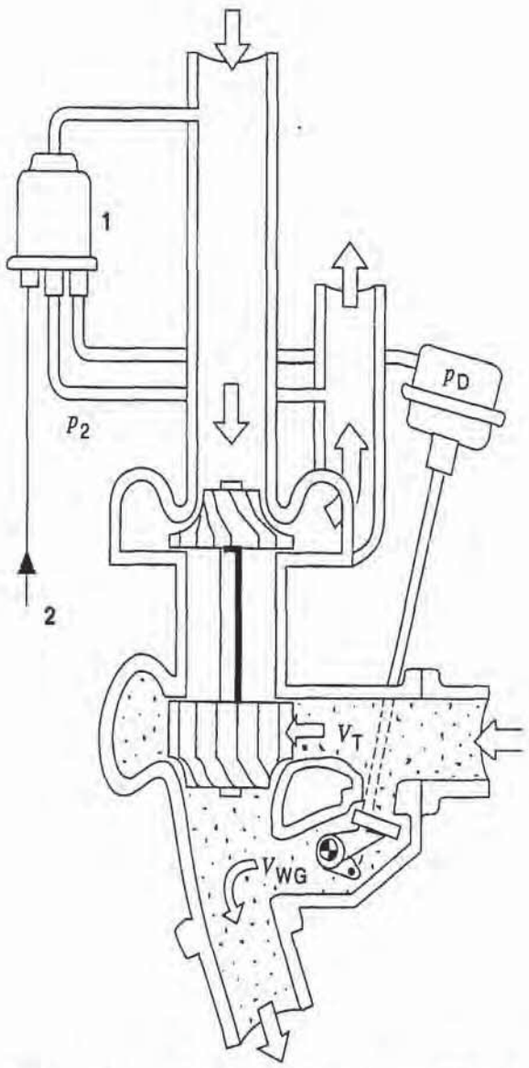


FIGURE 12.7 Electronic turbocharger boost control: solenoid valve (1), control signal from ECU (2), boost pressure (p_D), volume flow through turbine (V_T), volume flow through wastegate (V_{WG}).

ENGR & MATH SCI LIBRARY

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

The boost pressure control system is usually used in combination with knock control for turbocharged engines. When the ignition timing is retarded due to knock, an increase in already high exhaust temperatures for turbocharged engines occurs. To counteract the temperature increase, the boost pressure is reduced when the ignition timing is retarded past a predetermined threshold.

Engine/Vehicle Speed Control. Using the inputs of engine RPM and vehicle speed to the electronic control unit, thresholds can be established for limiting these variables with fuel cutoff. When the maximum speed is achieved, the fuel injectors are shut off. When the speed decreases below the threshold, fuel injection resumes.

EGR Control. By mixing a portion of the exhaust gas with the fresh intake air/fuel mixture, oxides of nitrogen (NO_x) can be reduced by lowering the peak combustion temperatures. However, the addition of exhaust gas can degrade driveability by causing combustion instability, especially at idle and low speeds and with a cold engine. The ECU references an engine RPM/load table of optimal EGR valve openings. The data table is developed on the engine dynamometer by analyzing the exhaust emissions. With increasing EGR, a point is reached where hydrocarbon (HC) emissions begin to increase. The optimal percent of EGR is just prior to that point.

The electronic control unit regulates a pneumatic- or solenoid-type valve to meter a certain quantity of exhaust gas back to the intake manifold. Typically, an engine coolant temperature threshold is also required before EGR is activated to avoid poor driveability. Under acceleration and at idle, EGR is deactivated.

Camshaft Control. There are two types of camshaft controls: phasing (i.e., overlap or intake/exhaust valve opening point) and valve lift and opening duration.

Camshaft Phasing Control. Valve overlap is a function of the rotation of the intake camshaft with respect to the exhaust camshaft. Overlap can be controlled by an electrohydraulic actuator. At idle and at high RPM, it is desirable to have the intake valves open and close later, which reduces the overlap. For idle, this reduces the residual exhaust gases that return with the fresh charge air and improves idle stability. At high RPM, late closing of the intake valve provides the best condition for maximum cylinder filling and, therefore, maximum output. For partial loads, a large valve overlap, where the intake opens early, is desirable. This allows for an increase in residual exhaust gas for improved exhaust emissions (Fig. 12.8).

Valve Lift and Opening Duration Control. Control of the valve lift and opening duration is accomplished by switching between two camshaft profiles. An initial cam specifies the optimal lift and duration for the low to middle RPM range. A second cam profile controls a higher valve lift and duration for high-RPM operation. By monitoring engine load and RPM, the ECU actuates the electrohydraulic device that switches from one cam profile to the other (Fig. 12.9).

Variable Intake Manifold Control. The goal of the engine design is to achieve the highest possible torque at low engine RPM as well as high output at high engine RPM. The torque curve of an engine is proportional to the air charge at any given engine speed. Therefore, a primary influence on the torque is the intake manifold geometric design. The simplest type of air charging uses the dynamics of the drawn-in air. The standard intake manifolds for multipoint engines consist of several intake runners and collectors converging at the throttle valve.

In general, short intake runners result in a high output at high RPM with a simultaneous loss of torque at low RPM. Long intake runners have the opposite effect. Due to intake valve and piston dynamics, pressure waves occur that oscillate within the intake manifold. Proper selection of runner lengths and collector sizes can result in the pressure waves arriving at the intake valves just before they are closing. This has a supercharging effect. The limitation of this method is that, for a given intake manifold configuration, the tuning peak can only occur at one operating point.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

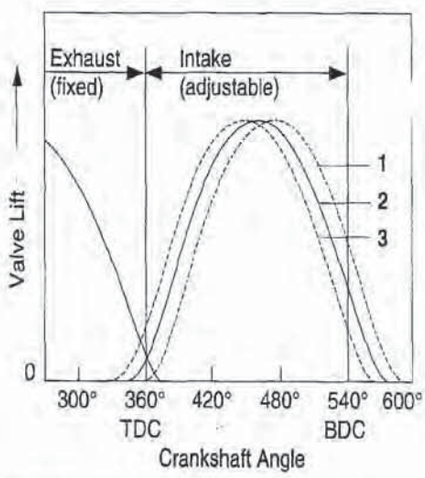


FIGURE 12.8 Adjustment angle for intake camshaft: retard (1), standard (2), advance (3).

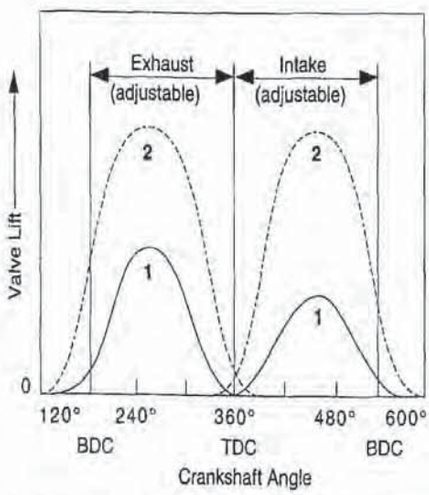


FIGURE 12.9 Selective camshaft lobe actuation: base cam lobe (1), auxiliary cam lobe (2).

Variable Intake Systems. To optimize the benefits of intake manifold charging, several systems have been developed that allow for changes in runner length and collector volume, depending on engine operating conditions. This allows for tuning peaks at more than one operating point. One method developed uses electronically controlled valves to close off areas of the intake manifold (Fig. 12.10). Inputs of engine load, RPM, and throttle angle determine the position of the valves.

12.2.2 Engine Control Modes

Engine Crank and Start. During engine cranking, the goal is to get the engine started with the minimal amount of delay. To accomplish this, fuel must be delivered that meets the requirements for starting for any combination of engine coolant and ambient temperatures. For a cold engine, an increase in the commanded air/fuel ratio is required due to poor fuel vaporization and "wall wetting," which decreases the amount of usable fuel. Wall wetting is the condensation of some of the vaporized fuel on the cold metal surfaces in the intake port and combustion chamber. It is critical that the fuel does not wet the spark plugs, which can reduce the effectiveness of the spark plug and prevent the plug from firing. Should plug wetting occur, it may be impossible to start the engine.

Fuel Requirement. Within the ECU ROM there are specific data tables to establish cold-start fuel based on engine coolant temperature. For two reasons, the lambda sensor output cannot be used during crank: the lambda sensor is below its minimum operating temperature and the air/fuel ratio required is outside the lambda sensor control window.

Many starting sequences use a front-loading strategy for fueling whereby the quantity of fuel is reduced after a speed threshold (RPM) is achieved, after a certain number of revolutions or at a defined time after the initial crank. Some systems also switch over from simultaneous injection to sequential injection after a speed threshold is achieved. For cold temperature starting, the fuel mixture may remain richer than $\lambda = 1$ after starting, due the continuing poor mixture formation in the cold induction system.

Ignition Timing Requirement. Ignition timing is controlled by the ECU during crank and is determined by engine coolant temperature and cranking speed. For a cold engine with low cranking speeds, ideal timing is near TDC. For higher cranking speeds, a slightly more advanced timing is optimal. Timing advance must be limited during cranking to avoid igniting

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

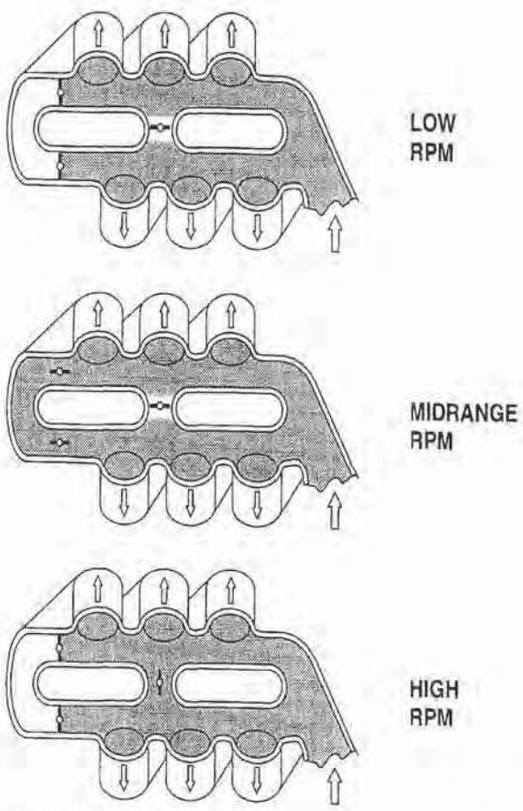


FIGURE 12.10 Variable configuration intake manifold.

the air/fuel mixture before the crankshaft reaches top dead center (TDC). A damaging torque reversal could occur that would damage the starter. After the engine starts, ignition timing is advanced to improve cold engine running as well as to reduce the need for fuel enrichment.

Engine Warm-Up. During the warm-up phase, there are three conflicting objectives: keep the engine operating smoothly (i.e., no stalls or driveability problems), increase exhaust temperature to quickly achieve operational temperature for catalyst (light-off) and lambda sensor so that closed-loop fuel control can begin operating, and keep exhaust emissions and fuel consumption to a minimum. The best method for achieving these objectives is very dependent on the specific engine application.

If the engine is still cold, fuel enrichment will be required to keep the engine running smoothly due, again, to poor fuel vaporization and wall wetting effects. The amount of enrichment is dependent on engine temperature and is a correction factor to the injector pulse width. This enrichment, combined with secondary air injection, also helps achieve the desired increase in catalyst temperature. To provide secondary air injection, an external air pump delivers fresh air downstream of the exhaust valves for a short time after start. The excess air causes oxidation (burning) of the excess HC and CO from the rich mixture in the exhaust manifold, which rapidly increases the temperature of the catalytic converter. The oxidation also removes harmful pollutants from the exhaust stream.

It is possible to increase the exhaust temperature by increasing the idle speed during warm-up. The increased idle speed may also be combined with a slightly retarded ignition timing, which increases temperatures in the exhaust, thereby promoting rapid warm-up of the catalyst.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Transient Compensation. During transitions such as acceleration or deceleration, the objective of the engine control system is to provide a smooth transition from one engine operating condition to another (i.e., no hesitations, stalls, bumps, or other objectionable driveability concerns), and keep exhaust emissions and fuel consumption to a minimum.

Acceleration Enrichment. When an increase in engine load and throttle angle occurs, a corresponding increase in fuel mixture richness is required to compensate for the increased wall wetting. The sudden increase in air results in a lean mixture that must be corrected swiftly to obtain good transitional response. The rate of change of engine load and throttle angle are used to determine the quantity of fuel during acceleration enrichment. The amount of fuel must be enough to provide the desired performance, but not so much as to degrade exhaust emissions and fuel economy.

During acceleration enrichment, the ignition timing is set for maximum torque without knocking. Additionally, when a large change in engine load occurs, some systems delay the ignition timing advance briefly to prevent engine knock, which may arise from a momentary lean mixture or from transient ignition timing errors.

Deceleration Enleanment. During deceleration modes, such as coasting or braking, there is no torque requirement. Therefore, the fuel may be shut off until either an increase in throttle angle is detected or the engine speed falls to a speed slightly above the idle RPM. Fuel shutoff or cutoff can decrease exhaust emissions by eliminating unburned HC and CO and may also improve fuel consumption. Fuel cutoff is also used to protect the catalytic converter from extreme high temperatures during extended overrun conditions. During transition to fuel cutoff, the ignition timing is retarded from its current setting to reduce engine torque and to assist in engine braking. The fuel is then shut off. During the transition, the throttle bypass valve or the main throttle valve may remain open for a short period to allow fresh air to oxidize the remaining unburned HC and CO to further reduce exhaust emissions. During development of the fuel cutoff strategy, the advantage of reduced emission effects and catalyst temperature control must be balanced against driveability requirements. The use of fuel cutoff may change the perceived amount of engine braking felt by the driver. In addition, care must be taken to avoid a "bump" feel when entering the fuel cutoff mode, due to the change in torque.

Full Load. Under steady state full-load conditions, such as for climbing a grade, it is desirable to control the air/fuel mixture and ignition timing to obtain maximum power and to also limit engine and exhaust temperatures. The best engine torque is typically delivered at about $\lambda = 0.9$ to 0.95. When the ECU determines the engine is operating at full load via the throttle valve sensor (at WOT), the commanded air/fuel mixture, if required, can be enriched. The lambda sensor signal cannot be used to provide correction to the air/fuel mixture because the rich operating point lies outside the lambda control window.

The ignition timing at full load is set to achieve the maximum torque without knocking. This initial value is determined through engine dynamometer testing. With a knock control system (see Sec. 12.2.1), the ignition timing is modified (retarded) when engine knock occurs. The modification required to eliminate the knock may be saved in the ECU so that the next time that engine RPM/load point occurs, less knocking will occur and less correction will be required.

Idle Speed Control. The objectives of the engine control system during idle are:

- Provide a balance between the engine torque produced and the changing engine loads, thus achieving a consistent idle speed even with various load changes due to accessories (i.e., air conditioning, power steering, and electrical loads) being turned on and off and during engagement of the automatic transmission. In addition, the idle speed control must be able to compensate for long-term changes in engine load, such as the reduction in engine friction that occurs with engine break-in.
- Provide the lowest idle speed that allows smooth running to achieve the lowest exhaust emissions and fuel consumption (up to 30 percent of a vehicle's fuel consumption in city driving occurs during idling).

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

To control the idle speed, the ECU uses inputs from the throttle position sensor, air conditioning, automatic transmission, power steering, charging system, engine RPM, and vehicle speed. There are currently two strategies used to control idle speed: air control and ignition control.

Air Control. The amount of air entering the intake manifold is controlled either by a bypass valve or by an actuator acting directly on the throttle valve. The bypass valve uses, for example, an electronically controlled motor controlled by the ECU that opens or closes a fixed amount. For large throttle valves, it may be desirable to use a bypass valve because a small change in throttle angle may result in a large change in air flow and, therefore, idle speed may be difficult to control. Using engine RPM feedback input, the ECU adjusts the air flow to increase or decrease the idle speed. A disadvantage to air control is that the response to load changes is relatively slow. To overcome this, air control is often combined with ignition timing control to provide acceptable idle speed control. The fuel quantity required at idle is determined by engine load and RPM. During closed-loop operation, this value is optimized by the lambda sensor closed-loop control.

Ignition Timing Control. Engine torque may be increased or decreased by advancing or retarding the ignition timing within an established window. This principle can be employed to help control idle speed. Ignition timing control is particularly desirable for responding to idle load changes because engine torque output changes more rapidly in response to a change in ignition timing than to a change in air valve position. Using the same inputs as for air control, the ECU adjusts the spark advance to either raise or lower the idle speed.

Anticipating Accessory Loads. Specific electric inputs to the ECU, such as a pressure switch located in the power steering system, are used to anticipate accessory loads so that the idle control system can compensate more quickly. This "feed forward" strategy allows better idle control than a strictly feedback system which does not respond until the idle speed begins to fall. When an accessory can be controlled by the ECU, further improvement in idle speed control is obtained. By delaying the load briefly after it is requested, the compensation sequence can begin before the load is actually applied. Such a load delay strategy is effective for controlling air conditioning compressor loads, for example. In this case, when the air conditioner is requested, the ECU begins to increase the idle speed first and then activates the A/C compressor.

12.2.3 Engine Control Diagnostics

The purpose of system diagnostics is to provide a warning to the driver when the control system determines that a malfunction of a component or system has occurred and to assist the service technician in identifying and correcting the failure (see Chap. 22). In many cases, to the driver, the engine may appear to be operating correctly, but excessive amounts of pollutants may be emitted. The ECU determines that a malfunction has occurred when a sensor signal received during normal engine operation or during a system test indicates there is a problem. For critical operations such as fuel metering and ignition control, if a required sensor input is faulty, a substitute value may be used by the ECU so that the engine will continue to operate, but likely not at optimal performance. It is also possible to apply an emergency measure if the failure of a component may result in engine or emission system damage. For example, if repeated misfires are detected in one cylinder, perhaps due to an ignition failure, the fuel injector feeding that cylinder can be shut off to avoid damage to the catalytic converter. When a failure occurs, the malfunction indicator light (MIL), visible to the driver, is illuminated. Information on the failure is stored in the ECU. A service technician can retrieve the information on the failure from the ECU and correct the problem.

Air Mass Sensor. For air mass measurement systems, the pulse width of the fuel injectors is calculated in the ECU from the air mass sensor input. As a comparison, the pulse width is also calculated from the throttle valve sensor and the engine RPM. If the pulse width values devi-

VW EX1015 Part II
U.S. Patent No. 6,557,540

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

ate by a predetermined amount, the discrepancy is stored in the ECU. Then, while the vehicle is being driven, plausibility tests determine which input is incorrect. When this has been determined, the appropriate failure code is saved in the ECU.

Misfire Detection. Misfiring is the lack of combustion in the cylinder. Misfiring can be caused by several factors including fouled or worn spark plugs, poor fuel metering, or faulty electrical connections. Even a small number of misfires may result in excessive exhaust emissions due to the unburned mixture. Increased misfire rates can damage the catalytic converter.

To determine if the engine is experiencing a misfire, the crankshaft speed fluctuation is monitored. If a misfire occurs, no torque is created during the power stroke of the cylinder(s) that is misfiring. A small decrease in the rotational speed of the crankshaft occurs. Because the change in speed is very small, highly accurate sensing of the crankshaft speed is required. In addition, a fairly complicated calculation process is required in order to distinguish misfiring from other influences on crankshaft speed. As was mentioned previously, if a cylinder repeatedly misfires, it is possible to shut off the fuel to that cylinder to prevent damage to the catalytic converter.

Catalytic Converter Monitoring. During the useful life of a catalytic converter, its efficiency decreases. If subjected to engine misfire, the decrease in efficiency occurs more rapidly. A loss in efficiency results in an increase in exhaust pollutants. For this reason, the catalytic converter is monitored. A properly operating catalytic converter transforms O_2, HC, CO, and NO_x into H_2O, CO_2, and N_2. The incoming air/fuel ratio oscillates from rich to lean due to the lambda closed-loop control strategy discussed in Sec. 12.2.1. Only a properly functioning catalytic converter is able to dampen these oscillations by storing and converting the incoming components. As the catalyst ages, this storage effect is diminished. To monitor the catalytic converter, an additional lambda sensor is installed downstream of the catalyst. The ECU compares the signal of the lambda sensor upstream with the lambda sensor downstream and determines if the catalytic converter is operating properly. If not, the ECU illuminates the malfunction indicator light (MIL) and stores a failure code.

Lambda Sensor Monitoring. To minimize exhaust emissions, the engine must operate within the catalytic converter window for air/fuel ratio (see Sec. 12.1.1 for a detailed description of the catalytic converter window). Output from the lambda sensor serves as feedback to the ECU to control the fuel within that window. When a lambda sensor is exposed to high heat for a long period of time, it may respond more slowly to changes in the air/fuel mixture. This can cause a deviation in the air/fuel mixture from the window, which would affect the exhaust emissions.

If the upstream lambda sensor operation is determined to be too slow, which can be detected by the system operation frequency, the ECU illuminates the malfunction indicator light (MIL) and a failure code is stored. Additionally, the ECU compares the output signal of the additional lambda sensor downstream of the catalytic converter with the lambda sensor signal upstream. With this, the ECU is able to detect deviations of the average value in air/fuel ratio.

For heated lambda sensors, the electric current and voltage of the heater circuit is monitored. To accomplish this, the heater is directly controlled by the ECU, not through a relay.

Fuel System Monitoring. To provide the correct air/fuel ratio, the ECU uses a preset data map with the optimal fuel required for each load and RPM point. The lambda closed-loop control system (see Sec. 12.2.1) provides feedback to the ECU on the necessary correction to the preset data points. The corrected information is stored in the ECU's RAM so that the next time that operating point is reached, less correction of the air/fuel ratio will be required. If the ECU correction passes a predetermined threshold, it is an indication that some component in the fuel supply system is outside of its operating range. Some examples are defective pressure regulator, defective manifold pressure sensor, intake system leakage, or exhaust system leakage. When the ECU determines a problem exists, the MIL is illuminated and a code is stored in the ECU.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Exhaust Gas Recirculation (EGR) Monitoring. There are currently two methods used to monitor EGR operation. One method confirms that hot exhaust gases are returning to the intake manifold during EGR operation by use of a temperature sensor in the intake manifold. The second method requires the EGR valve to be fully opened during coast operation, where high intake manifold vacuum occurs. The exhaust gas flowing into the manifold causes a measurable increase in pressure. Thus, if a measured increase in pressure does not occur, the EGR system is not operating.

Evaporative Emissions Control System (EECS) Monitoring. In general, a valve will be installed at the atmospheric side of the purge canister. During idle, this valve would close and the purge valve would open. Intake manifold vacuum would occur in the entire EECS. A pressure sensor in the fuel tank would provide a pressure profile during this test to the ECU, which would then determine if a leak existed in the system.

12.2.4 Fuel Delivery Systems

Overview. Fuel management in the spark ignition engine consists of metering the fuel, formation of the air/fuel mixture, transportation of the air/fuel mixture, and distribution of the air/fuel mixture. The driver operates the throttle valve, which determines the quantity of air inducted by the engine. The fuel delivery system must provide the proper quantity of fuel to create a combustible mixture in the engine cylinder. In general, two fuel delivery system configurations exist: *single-point* and *multipoint* (Fig. 12.11).

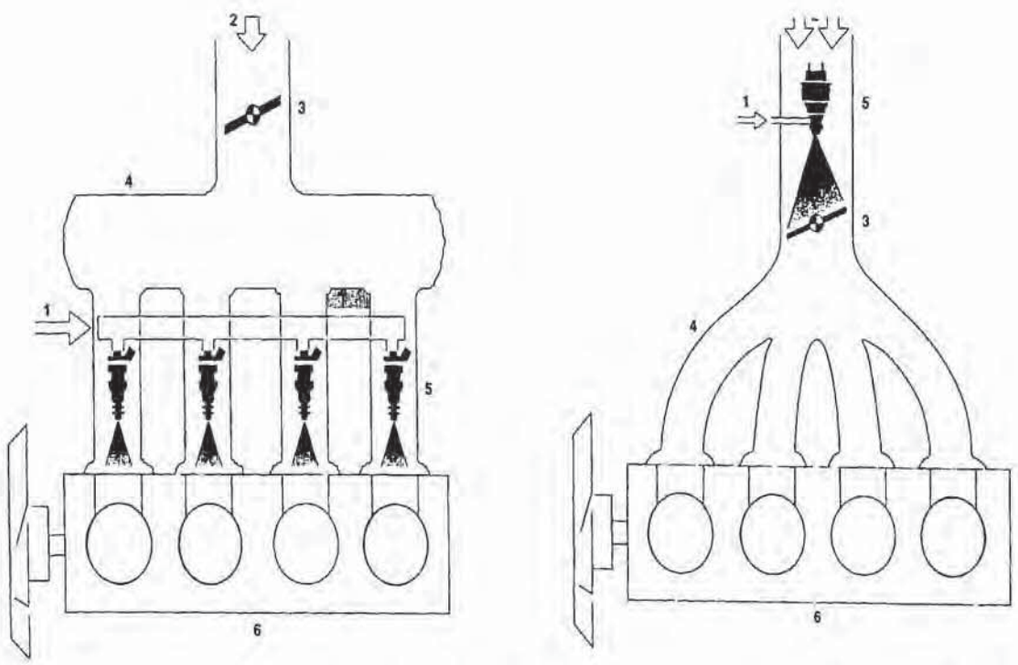


FIGURE 12.11 Air-fuel mixture preparation: right, single-point fuel injection; left, multipoint fuel injection with fuel (1), air (2), throttle valve (3), intake manifold (4), injector(s) (5), and engine (6).

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

For single-point systems such as carburetors or single-point fuel injection, the fuel is metered in the vicinity of the throttle valve. Mixture formation occurs in the intake manifold. Some of the fuel droplets evaporate to form fuel vapor (desirable) while others condense to form a film on the intake manifold walls (undesirable). Mixture transport and distribution is a function of intake manifold design. Uniform distribution under all operating conditions is difficult to achieve in a single-point system.

For multipoint systems, the fuel is injected near the intake valve. Mixture formation is supplemented by the evaporation of the fuel on the back of the hot intake valve. Mixture transport and distribution occurs only in the vicinity of the intake valve. The influence of the intake manifold design on uniform mixture distribution is minimized. Since mixture transport and distribution is not an issue, the intake manifold design can be optimized for air flow.

Single-Point Injection Systems A single-point injection system uses one or, in some cases, two electronic fuel injectors to inject fuel into the intake air stream. The main component is the fuel injection unit which is located upstream of the intake manifold.

Component Description. An electric fuel pump provides fuel at a medium pressure (typically 0.7 to 1.0 bar) to the electronic fuel injection unit (Fig. 12.12). The fuel injection unit houses the solenoid-operated fuel injector, which is located in the intake air flow above the throttle valve. This allows for homogeneous mixture formation and distribution. The injector spray pattern is designed to allow fuel to pass between the throttle valve and the throttle bore. To prevent vapor lock of the injector, fuel flows through the injector at all times. Fuel not used by the engine is returned to the fuel tank. The injector is activated in relation to the speed of the engine, typically once per ignition event. The length of the pulse width determines the quantity of fuel provided.

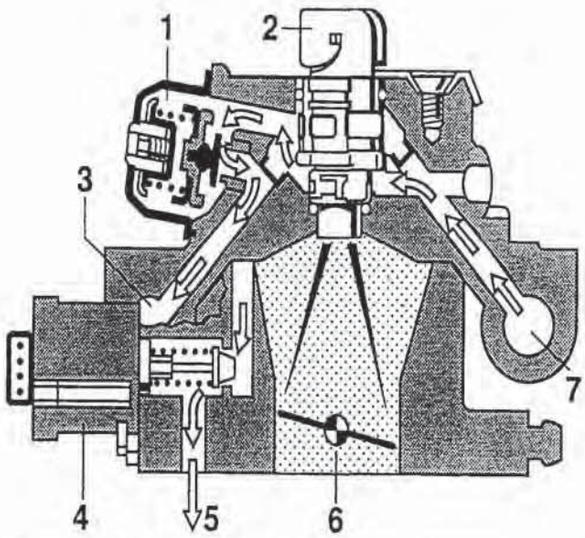


FIGURE 12.12 Single-point injection unit: pressure regulator (1), injector (2), fuel return (3), stepper motor for idle speed control (4), to intake manifold (5), throttle valve (6), and fuel inlet (7).

The electronic injection unit also houses the throttle position sensor and, in some cases, an inlet air temperature sensor which provides operating condition information to the ECU. The throttle valve actuator and fuel pressure regulator are also mounted on the injection unit. In addition, some units contain an air bypass valve for idle speed control. Engine temperature, battery voltage, and engine speed via the ignition system are all inputs to the ECU. The single-

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

point injection system also uses lambda closed-loop fuel control to optimize fuel metering within the lambda control window (see Sec. 12.2.1).

Adaptation to Operating Conditions. For cold-start and engine warm-up, the ECU uses engine temperature information to determine the correct amount of fuel and commands the fuel injector via a pulse width. Due to wall wetting and poor fuel vaporization when the engine is cold, an increase in mixture richness is required. As the engine warms up to operating temperature, the commanded pulse width is reduced.

During an acceleration transition, the ECU adds a correction factor (an increase) to the commanded injector pulse width. The sudden increase in air results in a lean mixture which must be corrected swiftly to obtain good transitional response. During a deceleration transition, the fuel can be shut off by simply not providing a pulse width signal to the injector to minimize exhaust emissions and fuel consumption.

During full-load operation, the air/fuel mixture can be enriched ($\lambda < 1$) to deliver maximum torque. The ECU determines full-load operation by the throttle position sensor (at or near wide-open throttle) and adds a correction to the injector pulse width to achieve the desired air/fuel mixture richness.

The single-point system can control the idle speed by ECU control of either a throttle valve actuator or a bypass valve. Idle speed is a function of engine operating temperature, whether the transmission is in drive, and what accessories are in use. Fuel metering at idle is determined by engine RPM and load as well as lambda closed-loop control.

Multipoint Fuel Injection Systems. A multipoint fuel injection system supplies fuel to each cylinder individually via a mechanical or solenoid-operated fuel injector located just upstream of the intake valve. Advantages of this system type compared to SPI systems are numerous:

- *Increased fuel economy.* On an SPI engine, due to the intake manifold configuration, mixture formation will differ at each cylinder. To provide adequate fuel for the leanest cylinder, too much fuel must be metered overall. In addition, during engine load changes, a film of fuel is deposited on the intake manifold walls. This leads to further variations in mixture from cylinder to cylinder. Multipoint injection provides the same quantity of fuel to each cylinder.
- *Higher power output.* With the fuel being injected near the intake valve, the rest of the intake manifold can be optimized for maximum air flow. The result is increased torque.
- *Improved throttle response.* Because the fuel is injected onto the intake valves, responses to increases in throttle position are swift. With an SPI system, the increased fuel required must travel the length of the intake manifold before entering the cylinder.
- *Lower exhaust emissions.* As was discussed for fuel economy, mixture variation in an SPI system creates increased exhaust emissions. Metering of the fuel at the intake valve decreases this variation. In addition, the system transport time is reduced, increasing the frequency at which the lambda closed-loop control system can switch air/fuel ratio. Catalytic converter efficiency is increased.

Although there are numerous advantages of the MPI systems over the SPI systems, there is still one important advantage the SPI systems have over the MPI systems. In general, SPI systems have better fuel preparation, similar to a carburetor.

Mechanically Controlled Continuous Injection System. This type of system meters the fuel as a function of the intake air quantity and injects it continuously onto the intake valves. This is accomplished by measuring the air flow as it passes through the air flow meter by means of deflection of a meter plate. The fuel is supplied through a fuel accumulator to the fuel distributor by an electric fuel pump. A primary-pressure regulator in the fuel distributor maintains constant fuel pressure. The fuel distributor, through its interface with the air flow meter and warm-up regulator, meters fuel to the continuously flowing fuel injectors.

Component Description

Mixture control unit. The mixture control unit houses the air flow meter and the fuel distributor. In the air flow meter, the measurement of the intake air serves as the basis for

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

determining the amount of fuel to be metered to the injectors. The air flow meter is located upstream of the throttle valve so that it measures all the air entering the engine. It consists of an air funnel, in which a sensor plate is free to pivot. Intake air flowing through the funnel causes a deflection of the sensor plate. The sensor plate is mechanically linked to a control plunger and movement of the plate results in movement of the control plunger. The control plunger movement determines the amount of fuel to be injected.

In the fuel distributor, the control plunger moves up and down in a cylindrically shaped device (barrel) with rectangular openings (metering slit), one for each engine cylinder. Increased air flow causes the control plunger to move upward, uncovering a larger area of the metering slit and increasing the fuel metered. Downstream of each metering slit is a differential-pressure valve that maintains a constant pressure drop across the metering slits at different flow rates. Due to the constant pressure, the fuel flow through the slits is directly proportional to the position of the control plunger (Fig. 12.13).

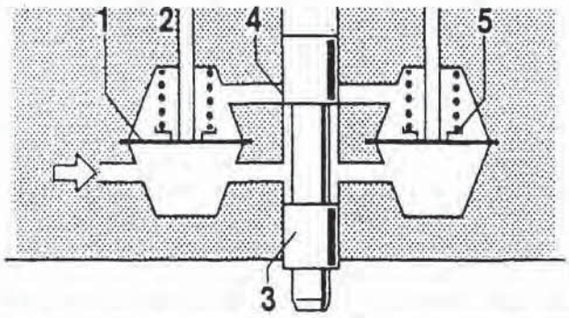


FIGURE 12.13 Fuel distributor for mixture control unit: diaphragm (1), to injector (2), control plunger (3), metering slot (4), differential pressure regulator (5).

Warm-up regulator. The warm-up regulator is used to richen the fuel mixture under cold engine conditions. It consists of a diaphragm valve and an electrically heated bimetallic spring. Under cold conditions, the warm-up regulator lowers the control pressure on the control plunger. The control pressure acts on the opposite end of the plunger from the air flow meter plate. A lower control pressure results in a lower force required to move the meter plate. Therefore, the same air flow causes the meter plate and control plunger to move a greater distance and additional fuel is metered to the injectors.

Fuel injectors. The injectors open at a pressure of approximately 3.6 bar. Atomization of the fuel occurs through oscillation (audible chatter) of the valve needle caused by the fuel flowing through it. The injectors remain open as long as fuel is provided above the opening pressure. Fuel is injected continuously into the intake port. When the intake valve opens, the mixture is drawn into the cylinder.

Auxiliary air valve. The auxiliary air valve provides additional air to the engine by bypassing the throttle valve during cold engine operation. This creates an increase in the idle speed needed during cold operation.

Thermo-time switch. The thermo-time switch controls the cold start valve as a function of time and engine temperature. Fuel enters the intake manifold from the cold start valve and further enriches the mixture to improve cold-starting at low ambient temperatures. When the engine is warm, the contacts in the thermo-time switch open and the cold-start valve is not used in starting the engine.

Lambda sensor. With the addition of a lambda sensor in the exhaust stream, a frequency valve, a modified fuel distributor, and an electronic control unit, the mechanically controlled fuel system can operate under lambda closed-loop control. The lambda sensor sig-

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

nal is read by the ECU. The ECU outputs electric pulses to an electromagnetic (frequency) valve. The frequency valve modulates the pressure to the lower chambers of the differential-pressure valves in the fuel distributor. This results in a modification of the pressure drop across the metering slits, effectively increasing or decreasing the amount of fuel injected. Figure 12.14 is a schematic of a typical mechanically controlled continuous injection system.

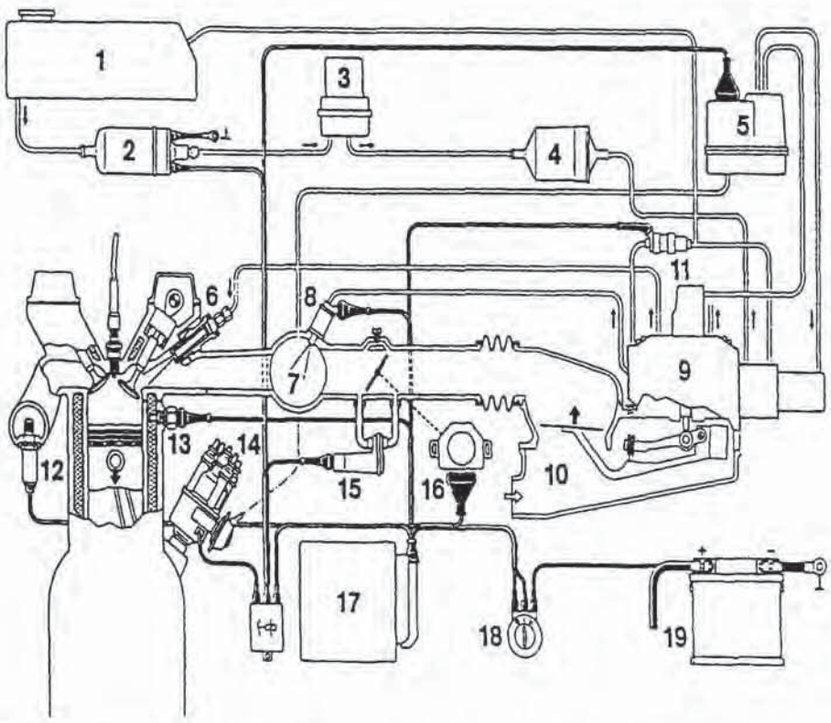


FIGURE 12.14 Schematic of mechanically controlled continuous injection system: fuel tank (1), electric fuel pump (2), fuel accumulator (3), fuel filter (4), warm-up regulator (5), injector (6), intake plenum (7), cold-start valve (8), fuel distributor (9), air flow sensor (10), electrohydraulic pressure actuator (11), lambda sensor (12), thermo-time switch (13), ignition distributor (14), auxiliary air valve (15), throttle switch (16), ECU (17), ignition switch (18), and battery (19).

Depending on the engine temperature, the cold-start valve injects extra fuel into the intake manifold for a limited period during cold start. The injection period is determined by a combination of time and temperature and is controlled by the thermo-time switch. As the engine temperature increases, this additional enrichment is no longer required and the thermo-time switch turns off the cold-start valve. For repeated start attempts or long cranking, the thermo-time switch turns off the cold-start injector after a given time. This minimizes engine flooding when engine start has not occurred.

As the engine continues to warm up, wall wetting and poor fuel vaporization still occur and mixture enrichment is still required until the engine reaches operating temperature. This enrichment is controlled as a function of temperature by the warm-up regulator. As the temperature increases, the warm-up regulator commands less and less additional fuel by increasing the control pressure.

For acceleration response, the air flow sensor "overswings" during quick throttle increases. This causes an additional quantity of fuel to be injected for acceleration enrichment. For full-load enrichment to achieve maximum power, a special warm-up regulator

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

that uses intake manifold pressure is required. At increased manifold pressures, i.e., during wide-open throttle, the warm-up regulator lowers the control pressure, which results in an increase in fuel delivery. Deceleration fuel shutoff is accomplished by diverting all intake air through an air bypass around the air flow sensor plate. With no air flow past the air flow sensor plate, the fuel pressure to the injectors is decreased below the opening pressure.

Idle speed for the cold-running engine is increased by the auxiliary air valve. The amount of additional air varies with engine temperature until the auxiliary air valve is closed and the idle speed is then controlled only be the air passing the throttle valve.

Electronically Controlled Continuous Injection. The basis of the electronically controlled continuous injection is still the mechanical hydraulic injection system discussed previously. This is supplemented by an electronic control unit (ECU) that allows for an increase in flexibility and the use of additional functions. This system incorporates additional sensors for detecting the engine temperature, the throttle valve position (load signal), and the air flow sensor plate deflection. This information is processed by the ECU, which then commands an electrohydraulic pressure actuator to adapt the injected fuel quantity for the present operating conditions.

In contrast to the mechanical system mentioned previously, the control pressure or counterpressure on the control plunger is not varied by a warm-up regulator. The control pressure remains constant and is the same as the primary pressure. The function of the warm-up regulator is now handled by the ECU and the electrohydraulic pressure actuator. Figure 12.15 is a schematic of a typical electronically controlled continuous injection system.

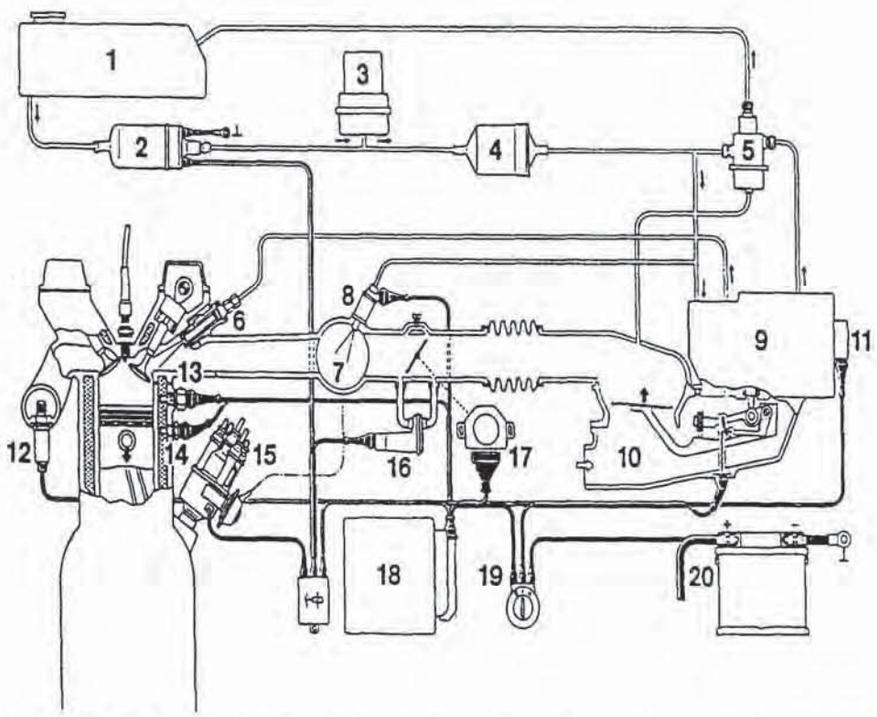


FIGURE 12.15 Schematic of an electronically controlled continuous fuel injection system: fuel tank (1), electric fuel pump (2), fuel accumulator (3), fuel filter (4), fuel pressure regulator (5), injector (6), intake plenum (7), cold-start valve (8), fuel distributor (9), air flow sensor (10), electrohydraulic pressure actuator (11), lambda sensor (12), thermo-time switch (13), coolant temperature sensor (14), ignition distributor (15), auxiliary air valve (16), throttle valve switch (17), ECU (18), ignition switch (19), and battery (20).

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Component description—electrohydraulic pressure actuator. The main difference in the componentry between the purely mechanical system and the electronically controlled system is the addition of the electrohydraulic actuator and the elimination of the warm-up regulator. With the addition of the ECU control of fuel metering, the purely mechanical warm-up regulator is no longer required. Depending on the signal received from the ECU, the electrohydraulic pressure actuator varies the pressure in the lower chambers of the differential pressure valves. This changes the amount of fuel delivered to the injectors.

Lambda closed-loop control. As with the mechanical system, the lambda sensor signal is processed by the ECU to determine mixture composition. The difference is that the ECU now commands the electrohydraulic actuator to modify the fuel metered, as opposed to the separate frequency valve, which is no longer necessary.

Adaptation to operating conditions. Depending on the engine temperature, the cold-start valve injects extra fuel into the intake manifold for a limited period during cold-start. The quantity to be injected is controlled by the ECU and is a function of engine temperature (from the engine temperature sensor). The thermo-time switch controls how long the cold-start valve remains active, depending on engine temperature and time.

Acceleration enrichment is controlled by the ECU. Input from the air flow sensor plate position sensor provides the ECU with information on how quickly the engine load has increased. The ECU commands additional enrichment via the electrohydraulic pressure actuator. For full-load enrichment for maximum power, the ECU receives input from the throttle position sensor that the throttle is wide open. The ECU then commands additional enrichment via the electrohydraulic pressure actuator. Deceleration fuel shutoff is also controlled by the ECU when the throttle valve switch indicates the throttle is closed and the engine speed is above idle RPM. The ECU signals the electrohydraulic pressure actuator to interrupt fuel delivery to the injectors.

Idle speed control can be a closed-loop function with the addition of the idle actuator valve. This valve is ECU-controlled and the RPM signal from the ignition, combined with the engine temperature signal, is used to determine its position for the correct idle speed.

Pulsed Fuel Injection Systems. Pulsed fuel injection systems are a further enhancement of the continuous injection systems. Today, most continuous injection systems have been replaced with pulsed fuel injection systems. Instead of injecting fuel continuously and controlling the quantity of fuel by modifying the delivery volume flow rate, the fuel quantity is controlled by the open time of the solenoid-operated injectors. The injectors are controlled directly by the ECU. For most systems, the fuel pressure drop across the injector, from the fuel rail to the intake manifold, is kept constant by using intake manifold air pressure to compensate the fuel pressure regulator. This type of system allows for still greater precision of fuel control and is usually coupled with an equally precise ignition timing control system.

Component description. Several multipoint pulsed injection systems exist in various configurations. The components discussed here serve as a general outline of this system type. Figure 12.16 is a schematic of a typical pulsed fuel injection system.

- *Inlet air sensing.* The inlet air charge can be measured directly using either an air flow meter or a mass air flow meter. The air flow meter is a vane-type meter, which uses the force of the incoming air to move a flap through a defined angle. The angular movement is converted by a potentiometer to a voltage ratio. The air flow meter requires an air inlet temperature sensor to correct for air density changes. The air mass flow meter measures the air mass directly by hot-wire or hot-film element. As the inlet air flow passes the heated element, a bridge circuit keeps the element at a constant temperature above the inlet temperature. The heating current required by the bridge circuit to maintain the element at a constant temperature is measured and converted to an air density value.

  The air charge can also be measured indirectly by measuring the inlet air temperature, intake manifold pressure, and engine RPM and then calculating the air charge (see Sec. 12.2.1 for further discussion on the calculation method which is called speed density air measurement).

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

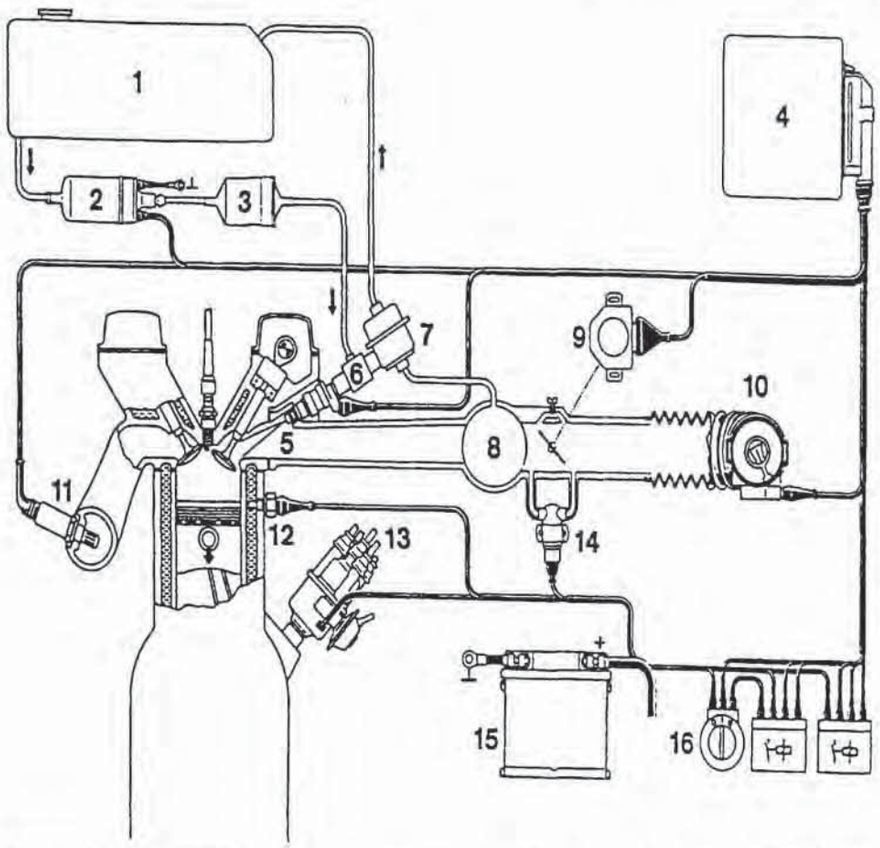


FIGURE 12.16 Schematic of a pulsed fuel injection system: fuel tank (1), electric fuel pump (2), fuel filter (3), ECU (4), injector (5), fuel distributor (6), fuel pressure regulator (7), intake plenum (8), throttle valve switch (9), hot-wire mass air flow sensor (10), lambda sensor (11), coolant temperature sensor (12), ignition distributor (13), idle speed actuator (14), battery (15), and ignition switch (16).

- *Fuel metering.* The fuel supply system includes an electric fuel pump, fuel filter, fuel rail, pressure regulator, and solenoid-operated injectors. The fuel pump provides more fuel than the maximum required by the engine. Fuel not used by the engine is returned to the fuel tank. The fuel rail supplies all injectors with an equal quantity of fuel and ensures the same fuel pressure at all injectors.

  The pressure regulator keeps the pressure differential across the injectors constant. It contains a diaphragm that has intake manifold pressure on one side and fuel rail pressure on the other. Normally, it is mounted at the outlet end of the fuel rail. The diaphragm operates a valve which opens at a differential pressure between 2.0 and 3.5 bar and allows excess fuel to return to the fuel tank.

  The fuel injectors are solenoid-operated valves that are opened and closed by means of electric pulses from the ECU. The injectors are mounted in the intake manifold and spray onto the back of the intake valves. In general, one injector is used for each cylinder.

  In addition, some systems also use a separate cold-start injector mounted in the intake manifold just downstream of the throttle valve. This injector ensures good fuel vaporization during cold-start and supplies the additional enrichment needed to start the cold engine. Control of the cold-start valve is either by the ECU directly or in conjunction with a thermo-time switch.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

- *Lambda closed-loop control.* The lambda sensor signal is processed by the ECU. The ECU determines the required injector pulse width to maintain the air/fuel ratio within the lambda control window (see Sec. 12.2.1 for further discussion on lambda closed-loop control).

Adaptation to operating conditions. For cranking, the fuel required is determined by a data table in the ECU with reference to engine temperature. The ECU then commands a pulse width for the fuel injectors. The air/fuel mixture is greatly enriched due to poor fuel vaporization and wall wetting, which reduces the amount of usable fuel. After start, the fuel mixture remains rich due to continuing poor air/fuel mixture formation. The amount of enrichment should be minimized to obtain good emission results. The target is to stay close to lambda (λ) = 1.

For acceleration enrichment, the throttle valve position sensor indicates that the throttle has moved rapidly. The ECU adds a correction factor to increase the pulse width so that a smooth transition occurs. For deceleration, the ECU uses input from the throttle position sensor and engine RPM to indicate that the throttle has closed and the engine speed is above the idle speed. Since no torque is required under this condition, the ECU provides no pulse width to the injectors and they are therefore closed. For full-load enrichment, when necessary, the ECU can provide an injector pulse width that would result in the engine achieving its maximum torque (roughly $\lambda = 0.9$). Fuel metering during idle is primarily controlled by lambda closed-loop control when the engine has reached operating temperature.

12.2.5 Ignition Systems

Overview. The purpose of the ignition system in the spark ignition engine is to initiate combustion of the air/fuel mixture by delivering a spark at precisely the right moment. The spark consists of an electrical arc generated across the electrodes of the spark plug. Two important factors for proper ignition are the energy of the spark and the point in the four-stroke cycle when the spark occurs (ignition timing).

Electrical Energy. The energy required for a spark to ignite an air/fuel mixture at stoichiometry depends on specific engine conditions. If there is insufficient energy available to ignite the air/fuel mixture, misfiring will occur. Misfiring will result in poor engine operation, high exhaust emissions, and possible catalytic converter damage. Therefore, the amount of ignition energy available must always exceed the amount necessary to ensure ignition even under adverse conditions.

Some of the conditions that affect ignitability of the air/fuel mixture are fuel atomization, access of the mixture to the spark, spark duration, and spark physical length. Fuel atomization is controlled by the fuel system and the engine design. Access to the spark depends on combustion chamber and spark plug design. Spark duration is a function of the ignition system. Spark physical length is determined by the spark plug dimensions (gap).

Ignition Timing. The ignition timing must be selected to meet the following objectives: maximize engine performance, limit fuel consumption, minimize engine knock, and minimize exhaust emissions. Unfortunately, all of these objectives cannot be achieved simultaneously under all operating conditions and compromises must be made.

It is desirable in the SI engine to have ignition of the combustible mixture occur prior to the piston reaching TDC on the compression stroke to achieve the best engine performance. The ignition spark must occur early enough to ensure that the peak cylinder combustion pressure occurs at the correct point after top dead center (ADC) under all operating conditions. Figure 12.17 is a graph of ignition angle vs. combustion pressure. The length of the combustion process from initial ignition to final combustion is approximately 2 ms. This combustion time remains relatively constant with respect to engine speed. Therefore, as the engine speed increases, the ignition spark must occur earlier in terms of crankshaft angle to ensure complete combustion.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

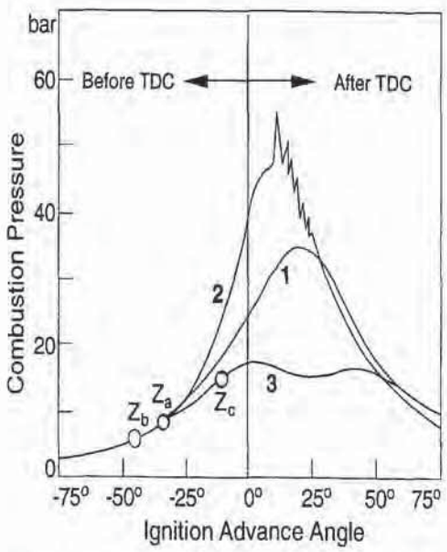


FIGURE 12.17 Combustion pressure curve for various ignition timing points: correct ignition advance Z_a (1), excessive ignition advance Z_b (2), and excessive ignition retard Z_c (3).

At low engine loads, the lower air charge and the residual gas content, due to valve overlap, serve to lengthen the time required for complete combustion. To compensate for this effect, the ignition timing is advanced at low loads to ensure that complete combustion occurs.

Ignition timing influences exhaust emissions and fuel consumption. With more advanced timing, the emission of unburned hydrocarbons (HC) and of oxides of nitrogen (NO_x) increases. Carbon monoxide (CO) emissions are not influenced greatly by ignition timing. To achieve improvements in fuel consumption, the air/fuel mixture must be lean. To ensure complete combustion for a lean mixture, the ignition timing must be advanced. However, as previously stated, advanced timing increases emissions of HC and NO_x.

Spark Ignition Systems. The general configuration of an ignition system consists of the following components: energy storage device, ignition timing mechanism, ignition triggering mechanism, spark distribution system, and spark plugs and high tension wires.

Inductive ignition systems use an ignition coil as the energy storage device. The coil also functions as a transformer, boosting the secondary ignition voltage. A typical turns-ratio of the primary to secondary winding is 1:100. Electrical energy is supplied to the coil's primary winding from the vehicle electrical system. Before the ignition point, the coil is charged during the dwell period to its interruption current. Open- or closed-loop dwell angle control ensures a sufficient interruption current even at high speeds. Sufficient ignition energy at the interruption current is ensured by an adequate coil design. At the ignition point, the primary current will be interrupted. The rapid change of the magnetic field induces the secondary voltage in the secondary winding. A distribution system assigns the high voltage to the corresponding spark plug. After exceeding the arcing over voltage at the spark plug, the coil will be discharged during the spark duration.

The ignition timing mechanism, ignition triggering mechanism, and the spark distribution system differ between ignition systems. Further discussion of these will occur within the discussion of each ignition system type.

The spark plugs provide the ignition energy via the high-tension wires to the air/fuel mixture in the cylinder to initiate combustion. The voltage required at the spark plug can be more

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

than 30 kV. Because the spark plug extends into the combustion chamber, it is exposed to extreme temperature and pressure conditions. Spark plug design and materials are chosen to ensure long-term operation under tough operating conditions.

A typical spark plug consists of a pair of electrodes, called a center and ground electrode, separated by a gap. The size of the gap is important and is specified for each plug type and engine. The center electrode is electrically connected to the top terminal of the plug which is attached to the high-tension wire. The electrical energy travels through the high-tension wire to the top terminal and down to the center electrode. The ground electrode is part of the threaded portion of the spark plug that is installed in the cylinder head. The ground electrode is at electrical ground potential because the negative terminal of the battery is also connected to the engine. The spark is produced when the high-voltage pulse travels to the center electrode and jumps the gap to the ground electrode.

Ignition System Types. Table 12.1 summarizes the various ignition systems used on SI engines.

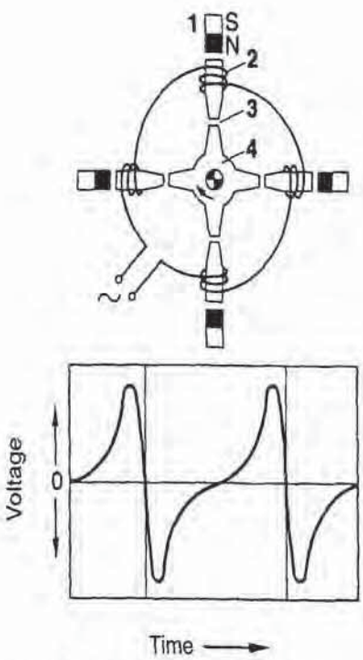


FIGURE 12.18 Induction-type pulse generator: permanent magnet (1), induction winding with core (2), variable air gap (3), trigger wheel (4).

Coil ignition. Breaker-triggered coil ignition systems have been replaced by breakerless transistorized ignition systems and are no longer installed as original equipment.

On breakerless transistorized ignition systems, the contact breaker's function is replaced by a magnetic pulse generator. The pulse generator is installed in the distributor and turns with the distributor shaft. There are commonly two types of pulse generators: induction-type and Hall-type. Induction-type pulse generators consist of a stator and a trigger wheel (Fig. 12.18). The stator consists of a permanent magnet, inductive winding, and core, and remains fixed. The trigger wheel teeth correspond to the number of cylinders, and the trigger wheel turns with the distributor shaft. The operating principle is that as the air gap changes between the stator and the rotor, the magnetic flux changes. The change in magnetic flux induces an ac voltage in the inductive winding. The frequency and magnitude of the alternating current increases with increasing engine speed. The electronic control unit or trigger box uses this information to trigger the ignition timing.

TABLE 12.1 Overview of Various Ignition Systems

	Ignition designation				
Ignition function	Coil system	Transistorized coil system	Capacitor discharge system	Electronic system with distributor	Electronic distributorless system
Ignition triggering	Mechanical	Electronic	Electronic	Electronic	Electronic
Ignition timing	Mechanical	Mechanical	Electronic	Electronic	Electronic
High-voltage generation	Inductive	Inductive	Capacitive	Inductive	Inductive
Spark distribution to appropriate cylinder	Mechanical	Mechanical	Mechanical	Mechanical	Electronic

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

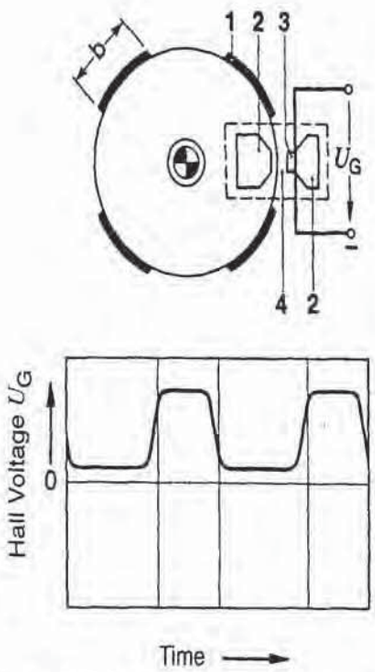


FIGURE 12.19 Hall-type pulse generator: vane (1), soft magnetic conductive elements (2), Hall IC (3), and air gap, UG-Hall sensor voltage (4).

Hall-type pulse generators utilize the Hall effect (Fig. 12.19). As the distributor shaft turns, the vanes of the rotor move through the air gap of the magnetic barrier. When the vane is not in front of the Hall IC, the sensor is subjected to a magnetic field. The magnetic flux density is high and thus the voltage U_G is at a maximum. As soon as the rotor vane enters the air gap, the magnetic flux runs through the vane area and is largely prevented from reaching the Hall layer. The voltage U_G is at a minimum. The resulting pulses switch the primary current off and on.

The distributor disburses the ignition pulses to the spark plugs via the high-tension wires in a specific sequence. It also adjusts the ignition timing by means of spark advance mechanisms. The distributor rotor is turned by the engine at one-half the crankshaft speed. The electrical energy is fed to the center of the rotor. While the rotor turns, the rotor electrode aligns with the outer electrodes that are connected to the high-tension wires. One outer electrode and high-tension wire connection exists for each cylinder. When alignment occurs between the center and outer electrode, the spark is distributed to that particular cylinder.

The spark advance mechanisms advance the ignition timing by rotating the distributor plate relative to the distributor shaft. The centrifugal advance increases the spark advance with increasing engine speed. The vacuum advance, using intake manifold vacuum, increases the spark advance at low engine speeds.

Capacitor discharge ignition system. The capacitive discharge system differs from the coil-type ignition systems previously discussed. Ignition energy is stored in the electrical field of a capacitor. Capacitance and charge voltage of the capacitor determine the amount of energy that is stored. The ignition transformer converts the primary voltage discharged from the capacitor to the required high voltage.

Electronic ignition—with distributor. Electronic ignition calculates the ignition timing electronically (Fig. 12.20). This replaces the function of the centrifugal advance and vacuum advance in the distributor discussed on the previous coil ignition systems. Because the ignition timing is not limited by mechanical devices, the optimal timing can be chosen for each engine operating point. Figure 12.21 is a comparison of an ignition map from a mechanical advance system and a map of an electronically optimized system. Also, additional influences such as engine knock detection can be used to modify the ignition timing. The engine speed input and crankshaft position input can be obtained from a sensor mounted near the crankshaft. Precision is improved over using the distributor-mounted trigger. This input is provided to the electronic control unit (ECU) along with the engine temperature and engine load. The ECU references data tables to determine the optimal spark advance for each engine operating condition. Additional corrections to the spark timing, such as for EGR usage or knock sensor detection, are made in the ECU.

Electronic ignition—distributorless. On distributorless ignition systems, the high voltage distribution is accomplished by using either single or double spark ignition coils. Ignition timing is determined by the ECU, as discussed for electronic ignition with distributor. For the double spark ignition coils, one coil exists for two corresponding cylinders. Two high-tension

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

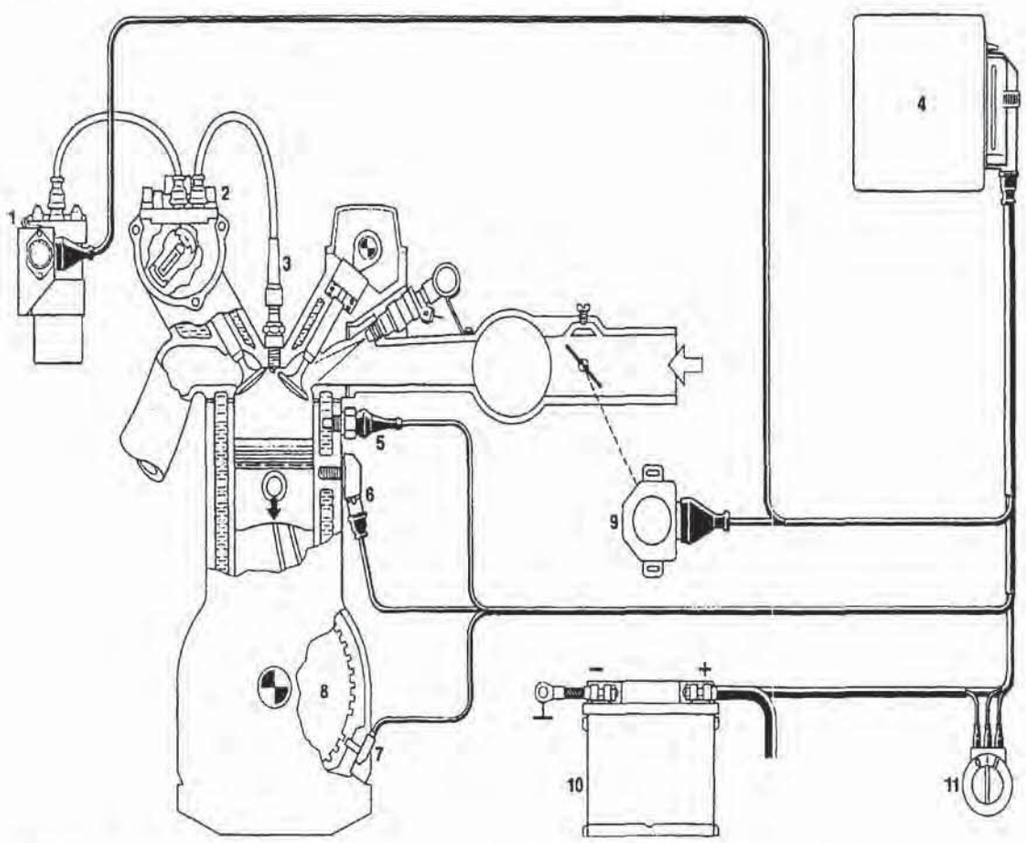


FIGURE 12.20 Schematic of an electronic ignition system with distributor: ignition coil (1), high-voltage distributor (2), spark plug (3), ECU (4), coolant temperature sensor (5), knock sensor (6), engine speed and crankshaft reference sensor (7), sensor wheel (8), throttle valve (9), battery (10), and ignition switch (11).

wires are routed from each coil to two cylinders, which are 360° out of phase. When the coil output stage is triggered via the ECU, a spark is delivered to both cylinders. One cylinder will be on the compression stroke, the other on the exhaust stroke. Because both cylinders are fired together, for a given crankshaft rotation, one will always be on the compression stroke and the other on the exhaust stroke. Therefore, there is no need to know which cylinder is compressing the ignitable mixture.

On single spark ignition coils, one coil exists for each cylinder. Each coil triggers only once during the four-stroke cycle. Because of this, it must be known which cylinder is on the compression stroke. Synchronization with the camshaft must occur. The information needed on camshaft position is supplied by a phase sensor mounted on the camshaft.

12.3 COMPRESSION IGNITION ENGINES

12.3.1 Engine Control Functions

Electronic engine controls are now being used on compression ignition (diesel) engines. These controls offer greater precision and control of fuel injection quantity and timing, engine speed, EGR, turbocharger boost pressure, and auxiliary starting devices. The following inputs

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

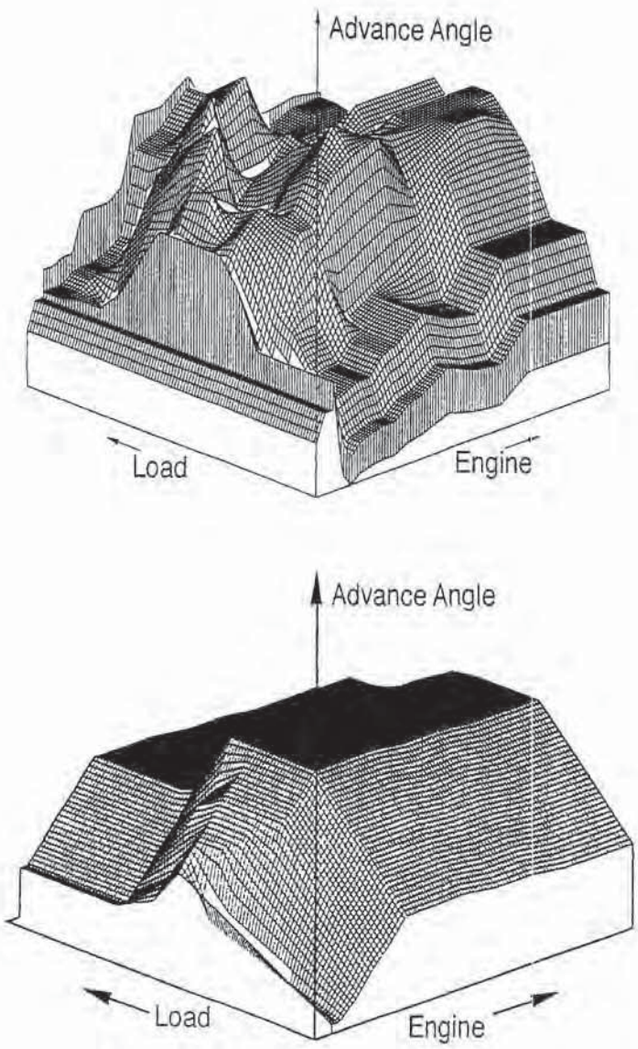


FIGURE 12.21 Ignition timing maps: electronically optimized (*above*) and mechanical advance system (*below*).

are used to provide the ECU with information on current engine operating conditions: engine speed; accelerator position; engine coolant, fuel, and inlet air temperatures; turbocharger boost pressure, vehicle speed, control rack, or control collar position (for control of fuel quantity); and amospheric pressure. Figure 12.22 is a schematic of an electronic engine control system on an in-line diesel fuel injection pump application.

Fuel Quantity and Timing. The fuel quantity alone controls a compression ignition engine's speed and load. The intake air is not throttled as in a spark ignition engine. The quantity of fuel to be delivered is changed by increasing or decreasing the length of fuel delivery time per injection. On the injection pump, the delivery time is controlled by the position of the control rack on in-line pumps and the position of the control collar on distributor-type pumps. An ECU-controlled actuator is used to move the control rack or the collar to increase or decrease the fuel delivery time. The ECU determines the correct length of delivery time (expressed as a function of control rack or collar position) using performance maps based on engine speed and calculated fuel quantity. Corrections and/or limitations as functions of

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

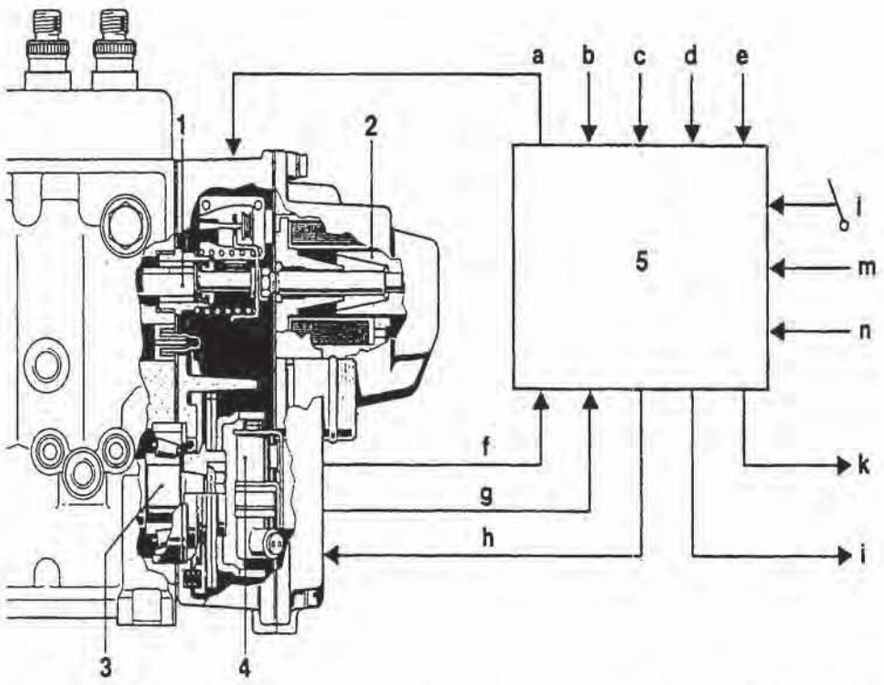


FIGURE 12.22 Electronic engine control system for an in-line injection pump: control rack (1), actuator (2), camshaft (3), engine speed sensor (4), ECU (5). Input/output: redundant fuel shutoff (a), boost pressure (b), vehicle speed (c), temperature—water, air, fuel (d), intervention in injection fuel quantity (e), speed (f), control rack position (g), solenoid position (h), fuel consumption and engine speed display (i), system diagnosis information (k), accelerator position (l), preset speed (m), and clutch, brakes, engine brake (n).

engine speed, temperature, and turbocharger boost pressure are used to modify the delivery time. In addition, the control rack or collar actuator contains a position sensor that provides feedback to the ECU on controller position. If the requested position differs from the commanded position, the ECU continues to move the controller via the actuator until the commanded and actual position are the same.

The start of injection time of the fuel at the cylinder is a function of the wave propagation speed (i.e., the speed of sound) of the fuel from the fuel injection pump to the injector. Because this time remains a constant, at increasing engine speed the delivery of fuel at the cylinder would be delayed with reference to crankshaft angle. Therefore, the timing at the injection pump must be advanced with increasing engine speed so that the start of injection occurs at the same crankshaft angle at higher engine speeds. Selection of injection timing has a large impact on exhaust emissions and engine noise. Delaying the start of injection reduces NO_x emissions, but excessive delay increases HCs in the exhaust. A 1° deviation in injection timing can increase NO_x emissions by 5 percent and HC emissions by as much as 15 percent. Therefore, precise control of the start of injection is essential.

Although many systems use mechanical devices to control injection timing, electronic control of injection timing is being used on some pump types. The advantage of electronic control is that a sophisticated timing data map can be used that provides the best injection timing for exhaust emissions under various operating conditions. On electronic control systems, the start of injection is monitored at the injector nozzle by a needle-motion sensor. The ECU uses this information to determine and control the injection timing. The timing is then modified by control of a pulse-width modulated solenoid valve. The valve varies the pressure exerted on the spring-loaded timing device plunger. The plunger rotates the pump's collar ring (for distributor type pumps) in the opposite direction of the pump's rotation which advances the timing.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Speed Control. As was mentioned previously, for a CI engine, fuel quantity alone controls the engine's speed and load. Therefore, presuming adequate injected fuel quantity, an unloaded CI engine can speed up out of control and destroy itself. Because of this, a governor is required to limit the engine's maximum speed. In addition, governors are also used for low idle and cruise control to maintain a constant engine or vehicle speed and meter the correct fuel for cold-starting. Fuel is also controlled as a function of speed and boost pressure to limit smoke levels, engine torque, and exhaust gas temperatures. On an electronically controlled CI engine, the governor's functions are controlled by the fuel delivery system described previously. Engine speed is provided by an RPM sensor that monitors the periods of angular segments between the reference marks on the engine's flywheel or in the in-line injection pump.

EGR Control. Rerouting of exhaust gases into the intake air stream is known as exhaust gas recirculation (EGR). EGR reduces the amount of oxygen in the fresh intake charge while increasing its specific heat. This lowers combustion temperatures and results in lower NO_x emissions. However, excessive amounts of EGR result in higher emissions of soot (particulates), CO, and HCs all due to insufficient air. Also, the introduction of EGR can have an adverse affect on driveability during cold-engine operation, full-load operation, and at idle. It is best, therefore, to control the EGR valve with the ECU. Both pneumatically controlled and solenoid-controlled EGR valves are in use. The ECU determines when and how much EGR will occur based on engine temperature and accelerator position.

Turbocharger Boost Pressure Control. Engines that have turbochargers benefit significantly from electronic boost pressure control. If only a pneumatic-mechanical wastegate is used, only one boost pressure point for the entire operating range is used to divert the exhaust gas away from the turbine side of the turbocharger. This creates a compromise for part-load conditions because all the exhaust gases must pass the turbine. The result is increased exhaust backpressure, more turbocharger work, more residual exhaust gas in the cylinders, and higher charge air temperatures.

By controlling the wastegate with a pulse-width-modulated solenoid valve, the wastegate can be opened at different pressures depending on the engine operating conditions. Therefore, only the level of air charge pressure required is developed. The electronic control unit uses information on engine speed and accelerator position to reference a data table and the proper boost pressure (actually, duty cycle of the control valve) is determined. On systems using intake manifold pressure sensors, a closed-loop control system can be developed to compare the specified value with the measured value.

Glow Plug Control. Electronic control of the glow plug duration can be handled by the ECU or a separate control unit. Input for determining glow time is from an engine coolant temperature sensor. At the end of the specified glow period, the controller turns out the start indicator light to signal the driver that the engine can be started. The glow plugs remain energized while the starter is engaged. An engine load monitor is used to switch off the glow process after start. To limit the loads on the battery and the glow plugs, a safety override is also used.

12.3.2 Fuel Delivery Systems

The diesel fuel delivery system comprises a low- and high-pressure side. On the low-pressure side is the fuel tank, fuel filter, fuel supply pump, overflow valve, and fuel supply lines. The high-pressure side is initiated in the plunger and barrel assembly and continues through the delivery valve, high-pressure injection lines, and injection nozzle.

The fuel injection pump must deliver fuel at a pressure between 350 and 1200 bar, depending on the engine's combustion configuration. The quantity and timing of injection must be precisely controlled to achieve good mixture quality and to minimize exhaust emissions.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

Fuel Injection Process. An engine-driven camshaft (in-line pump) or cam plate (distributor pump) drives the injection pump's plunger in the supply direction, creating pressure in the high-pressure gallery. The delivery valve responds to the increase in pressure by opening. This sends a pressure wave to the injection nozzle at the speed of sound. The needle valve in the nozzle overcomes the spring force of the injection nozzle spring and lifts from its seat when the opening pressure is reached. Fuel is then injected from the spray orifices into the engine's combustion chamber. The injection process ends with the opening of the spill port in the plunger and barrel assembly. This causes the pressure in the pump chamber to collapse, which then causes the delivery valve to close. Due to the action of the delivery valve relief collar, the pressure in the injection line is reduced to the "stand-by pressure." The stand-by pressure is determined to ensure that the injector nozzle closes quickly to eliminate fuel dribble, and the residual pressure waves in the lines prevent the nozzles from reopening.

ABOUT THE AUTHORS

GARY C. HIRSCHLIEB is chief engineer, engine management systems, for the Robert Bosch Corp. He previously held various engineering and sales responsibilities with Bosch. In his earlier career, he worked as a senior engineer in powertrain development for Ford tractor operations, and as a sales engineer with GTE, and as an engineer in plant engineering for GM Truck and Coach.

GOTTFRIED SCHILLER is engineering manager, engine management systems, for Robert Bosch Corp. His previous positions with Bosch included applications engineering, engine management systems; application engineer, diesel systems; and development engineer, diesel products.

SHARI STOTTLER is now a self-employed technical writer, but, until 1993, she was a senior application engineer with Robert Bosch Corp. Prior to that she had been an engineering project coordinator for Honda of North America, Manufacturing, and a product engineer with General Motors Corp.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

CHAPTER 13
TRANSMISSION CONTROL

Kurt Neuffer, Wolfgang Bullmer, and Werner Brehm
Robert Bosch GmbH

13.1 INTRODUCTION

In North America and Japan, 80 to 90 percent of all passenger cars sold have automatic transmissions (ATs), but in Europe only 10 to 15 percent of passenger cars sold have ATs. There are two main reasons for the difference. In Europe, drivers tend to view ATs, compared to manual transmissions, as detrimental to driveability and responsible for a somewhat higher fuel consumption. But implementation of electronic control concepts has invalidated both of those arguments.

Since the introduction of electronic transmission controls units (TCUs) in the early 1980s by Renault and BMW (together with a four-speed transmission from Zahnradfabrik Friedrichshafen, or ZF), the acceptance of the AT rose steeply, even in Europe. For this reason, all new ATs are designed with electronic control. The market for ATs is divided into stepped and continuously variable transmissions (CVTs). For both types the driver gets many advantages. In stepped transmissions, the smooth shifts can be optimized by the reduction of engine torque during gear shift, combined with the correctly matched oil pressure for the friction elements (clutches, brake bands). The reduction of shift shocks to a very low or even to an unnoticeable level has allowed the design of five-speed ATs where a slightly higher number of gear shifts occur. In today's standard systems, the driver can choose between sport and economic drive programs by operating a selector switch. In highly sophisticated newer systems, the selection can be replaced by the self-adaptation of shift strategies. This leads not only to better driveability but also to a significant reduction in fuel consumption. Additionally, a well-matched electronic control of the torque converter lockup helps to improve the yield of the overall system. Both automotive and transmission manufacturers benefit from the reduced expense resulting from the application of different car/engine combinations. Different shift characteristics are easy to implement in software, and much adaptation can be achieved by data change, leaving the transmission hardware and TCU unchanged. The reduction of power losses in friction elements increases the life expectancy and enables the optimization of transmission hardware design.

With the CVT, one of the biggest obstacles to the potential reduction in fuel consumption by operating the engine at its optimal working point is the power loss from the transmission's oil pump. Only with electronic control is it possible to achieve the required yield by matching the oil mass-stream and oil pressure for the pulleys to the actual working conditions.

To guarantee the overall economic solution for an electronically controlled transmission, either stepped or CVT, the availability of precision electrohydraulic actuators is imperative.

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540

13.2 SYSTEM COMPONENTS

The components of an electronic transmission control system are a transmission which is adapted to the electronic control requirements and an electronic control unit with corresponding inputs and outputs and attached sensor elements.

13.2.1 Transmission

The greatest share of electronically controlled transmissions currently on the market consists of four- or five-speed units with a torque converter lockup clutch, commanded by the control unit. Market share for five-speed transmissions is continuously increasing. With electronically controlled transmissions there are numerous possibilities to substitute mechanical and hydraulic components with electromechanical or electrohydraulic components. One basic method is to substitute only the shift point control. In a conventional pure hydraulic AT, the gear shifts are carried out by mechanical and hydraulic components. These are controlled by a centrifugal governor that detects the vehicle speed, and a wire cable connected to the throttle plate lever. With an electronic shift point control, on the other hand, an electronic control unit detects and controls the relevant components. In the transmission's hydraulic control unit, mechanical and hydraulic components are replaced by electrohydraulic controlling elements, usually in the form of electrohydraulic on/off solenoids. This way the number of solenoids, as well as the control logic, can be varied over a wide range. For example, for each gear, one specific solenoid can operate the relevant clutch for this gear shift. Alternatively, there can be one solenoid for each gear change, which is switched corresponding to the shift command. In this way, only three solenoids are required in a four-speed transmission. In some current designs, the gears are controlled by a logical combination of solenoid states. This design needs only two gear-controlling solenoids for a four-speed transmission. For five-speed applications, accordingly, three solenoids are required (Table 13.1)

TABLE 13.1 Example of a Gear-Solenoid Combination for a Five-Speed Transmission Application

	Solenoid 1	Solenoid 2	Solenoid 3
1st gear	on	on	on
2nd gear	on	on	off
3rd gear	on	off	off
4th gear	off	off	off
5th gear	off	on	off

The hydraulic pressure is controlled in this basic application by a hydraulic proportional valve which is, in turn, controlled by a wire cable connected to the throttle plate lever. With this design, the shift points can be determined by the electronic TCU, resulting in a wide range of freely selectable driving behaviors regarding the shift points. It is also possible to use different shift maps according to switch or sensor signals. The influence on driving comfort during gear shifting in this electronic transmission control application has important restrictions. The only possible way to control shift smoothness is with an interface to the electronic engine management. This way, the engine output torque is influenced during gear shifting. A systematic wide-range control of the hydraulic pressure during and after the gear shift necessitates the replacement of the hydraulic pressure governor with an electronically controlled hydraulic solenoid. This design allows the use of either a pulse-width-modulated (PWM) solenoid or a pressure regulator. The choice of which type of pressure control solenoid to use results from the requirements concerning shift comfort under all driving conditions. For

PETITIONER'S EXHIBIT 1002
US PATENT NO. 6,557,540