2 FREQUENCY HOPPING SAMPLE DATA

The section contains three sets of sample data showing the basic and adapted hopping schemes for different combinations of addresses and initial clock values.

Sample Data

2.1 FIRST SET

Hop sequence {k} for PAGE SCAN/INQUIRY SCAN SUBSTATE:
CLKN start: 0x0000000
UAP / LAP: 0x00000000

#ticks:	0000	1000	2000	3000	4000	5000	6000	7000
0x0000000:	0	2	4	6	8	10	12	14
0x0008000:	16	18	20	22	24	26	28	30
0x0010000:	32	34	36	38	40	42	44	46
0x0018000:	48	50	52	54	56	58	60	62
0x0020000:	0	2	4	6	8	10	12	14
0x0028000:	16	18	20	22	24	26	28	30
0x0030000:	32	34	36	38	40	42	44	46
0x0038000:	48	50	52	54	56	58	60	62

Hop sequence {k} for PAGE STATE/INQUIRY SUBSTATE:
CLKE start: 0x0000000
UAP / LAP: 0x00000000

#ticks:	00 01 02 03	04 05 06 07	08 09 0a 0b	0c 0d 0e 0f
0x0000000:	48 50 09 13	52 54 41 45	56 58 11 15	60 62 43 47
0x0000010:	00 02 64 68	04 06 17 21	08 10 66 70	12 14 19 23
0x0000020:	48 50 09 13	52 54 41 45	56 58 11 15	60 62 43 47
0x0000030:	00 02 64 68	04 06 17 21	08 10 66 70	12 14 19 23
...				
0x0001000:	48 18 09 05	20 22 33 37	24 26 03 07	28 30 35 39
0x0001010:	32 34 72 76	36 38 25 29	40 42 74 78	44 46 27 31
0x0001020:	48 18 09 05	20 22 33 37	24 26 03 07	28 30 35 39
0x0001030:	32 34 72 76	36 38 25 29	40 42 74 78	44 46 27 31
...				
0x0002000:	16 18 01 05	52 54 41 45	56 58 11 15	60 62 43 47
0x0002010:	00 02 64 68	04 06 17 21	08 10 66 70	12 14 19 23
0x0002020:	16 18 01 05	52 54 41 45	56 58 11 15	60 62 43 47
0x0002030:	00 02 64 68	04 06 17 21	08 10 66 70	12 14 19 23
...				
0x0003000:	48 50 09 13	52 22 41 37	24 26 03 07	28 30 35 39
0x0003010:	32 34 72 76	36 38 25 29	40 42 74 78	44 46 27 31
0x0003020:	48 50 09 13	52 22 41 37	24 26 03 07	28 30 35 39
0x0003030:	32 34 72 76	36 38 25 29	40 42 74 78	44 46 27 31

Hop sequence {k} for SLAVE PAGE RESPONSE SUBSTATE:
CLKN* = 0x0000010
UAP / LAP: 0x00000000

#ticks:	00	02 04	06 08	0a 0c	0e 10	12 14	16 18	1a 1c	1e
0x0000012:	64	02 68	04 17	06 21	08 66	10 70	12 19	14 23	16
0x0000032:	01	18 05	20 33	22 37	24 03	26 07	28 35	30 39	32
0x0000052:	72	34 76	36 25	38 29	40 74	42 78	44 27	46 31	48
0x0000072:	09	50 13	52 41	54 45	56 11	58 15	60 43	62 47	00

Hop sequence {k} for MASTER PAGE RESPONSE SUBSTATE:
Offset value: 24
CLKE* = 0x0000012
UAP / LAP: 0x00000000

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000014:	02 68	04 17	06 21	08 66	10 70	12 19	14 23	16 01
0x0000034:	18 05	20 33	22 37	24 03	26 07	28 35	30 39	32 72
0x0000054:	34 76	36 25	38 29	40 74	42 78	44 27	46 31	48 09
0x0000074:	50 13	52 41	54 45	56 11	58 15	60 43	62 47	00 64

Sample Data

Hop sequence {k} for CONNECTION STATE (Basic channel hopping sequence; ie, non-AFH):
CLK start: 0x0000010
UAP/LAP: 0x00000000

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010:	08 66	10 70	12 19	14 23	16 01	18 05	20 33	22 37
0x0000030:	24 03	26 07	28 35	30 39	32 72	34 76	36 25	38 29
0x0000050:	40 74	42 78	44 27	46 31	48 09	50 13	52 41	54 45
0x0000070:	56 11	58 15	60 43	62 47	32 17	36 19	34 49	38 51
0x0000090:	40 21	44 23	42 53	46 55	48 33	52 35	50 65	54 67
0x00000b0:	56 37	60 39	58 69	62 71	64 25	68 27	66 57	70 59
0x00000d0:	72 29	76 31	74 61	78 63	01 41	05 43	03 73	07 75
0x00000f0:	09 45	13 47	11 77	15 00	64 49	66 53	68 02	70 06
0x0000110:	01 51	03 55	05 04	07 08	72 57	74 61	76 10	78 14
0x0000130:	09 59	11 63	13 12	15 16	17 65	19 69	21 18	23 22
0x0000150:	33 67	35 71	37 20	39 24	25 73	27 77	29 26	31 30
0x0000170:	41 75	43 00	45 28	47 32	17 02	21 04	19 34	23 36
0x0000190:	33 06	37 08	35 38	39 40	25 10	29 12	27 42	31 44
0x00001b0:	41 14	45 16	43 46	47 48	49 18	53 20	51 50	55 52
0x00001d0:	65 22	69 24	67 54	71 56	57 26	61 28	59 58	63 60
0x00001f0:	73 30	77 32	75 62	00 64	49 34	51 42	57 66	59 74
0x0000210:	53 36	55 44	61 68	63 76	65 50	67 58	73 03	75 11
0x0000230:	69 52	71 60	77 05	00 13	02 38	04 46	10 70	12 78
0x0000250:	06 40	08 48	14 72	16 01	18 54	20 62	26 07	28 15
0x0000270:	22 56	24 64	30 09	32 17	02 66	06 74	10 19	14 27
0x0000290:	04 70	08 78	12 23	16 31	18 03	22 11	26 35	30 43
0x00002b0:	20 07	24 15	28 39	32 47	34 68	38 76	42 21	46 29
0x00002d0:	36 72	40 01	44 25	48 33	50 05	54 13	58 37	62 45
0x00002f0:	52 09	56 17	60 41	64 49	34 19	36 35	50 51	52 67
0x0000310:	38 21	40 37	54 53	56 69	42 27	44 43	58 59	60 75
0x0000330:	46 29	48 45	62 61	64 77	66 23	68 39	03 55	05 71
0x0000350:	70 25	72 41	07 57	09 73	74 31	76 47	11 63	13 00
0x0000370:	78 33	01 49	15 65	17 02	66 51	70 67	03 04	07 20
0x0000390:	68 55	72 71	05 08	09 24	74 59	78 75	11 12	15 28
0x00003b0:	76 63	01 00	13 16	17 32	19 53	23 69	35 06	39 22
0x00003d0:	21 57	25 73	37 10	41 26	27 61	31 77	43 14	47 30
0x00003f0:	29 65	33 02	45 18	49 34	19 04	21 08	23 20	25 24

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with all channel used; ie, AFH(79)):

CLK start: 0x0000010

ULAP: 0x00000000

Used Channels:0x7fffffffffffffffffff

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	08 08	10 10	12 12	14 14	16 16	18 18	20 20	22 22
0x0000030	24 24	26 26	28 28	30 30	32 32	34 34	36 36	38 38
0x0000050	40 40	42 42	44 44	46 46	48 48	50 50	52 52	54 54
0x0000070	56 56	58 58	60 60	62 62	32 32	36 36	34 34	38 38
0x0000090	40 40	44 44	42 42	46 46	48 48	52 52	50 50	54 54
0x00000b0	56 56	60 60	58 58	62 62	64 64	68 68	66 66	70 70
0x00000d0	72 72	76 76	74 74	78 78	01 01	05 05	03 03	07 07
0x00000f0	09 09	13 13	11 11	15 15	64 64	66 66	68 68	70 70
0x0000110	01 01	03 03	05 05	07 07	72 72	74 74	76 76	78 78
0x0000130	09 09	11 11	13 13	15 15	17 17	19 19	21 21	23 23
0x0000150	33 33	35 35	37 37	39 39	25 25	27 27	29 29	31 31
0x0000170	41 41	43 43	45 45	47 47	17 17	21 21	19 19	23 23
0x0000190	33 33	37 37	35 35	39 39	25 25	29 29	27 27	31 31
0x00001b0	41 41	45 45	43 43	47 47	49 49	53 53	51 51	55 55
0x00001d0	65 65	69 69	67 67	71 71	57 57	61 61	59 59	63 63
0x00001f0	73 73	77 77	75 75	00 00	49 49	51 51	57 57	59 59
0x0000210	53 53	55 55	61 61	63 63	65 65	67 67	73 73	75 75
0x0000230	69 69	71 71	77 77	00 00	02 02	04 04	10 10	12 12
0x0000250	06 06	08 08	14 14	16 16	18 18	20 20	26 26	28 28
0x0000270	22 22	24 24	30 30	32 32	02 02	06 06	10 10	14 14
0x0000290	04 04	08 08	12 12	16 16	18 18	22 22	26 26	30 30
0x00002b0	20 20	24 24	28 28	32 32	34 34	38 38	42 42	46 46
0x00002d0	36 36	40 40	44 44	48 48	50 50	54 54	58 58	62 62
0x00002f0	52 52	56 56	60 60	64 64	34 34	36 36	50 50	52 52
0x0000310	38 38	40 40	54 54	56 56	42 42	44 44	58 58	60 60
0x0000330	46 46	48 48	62 62	64 64	66 66	68 68	03 03	05 05
0x0000350	70 70	72 72	07 07	09 09	74 74	76 76	11 11	13 13
0x0000370	78 78	01 01	15 15	17 17	66 66	70 70	03 03	07 07
0x0000390	68 68	72 72	05 05	09 09	74 74	78 78	11 11	15 15
0x00003b0	76 76	01 01	13 13	17 17	19 19	23 23	35 35	39 39
0x00003d0	21 21	25 25	37 37	41 41	27 27	31 31	43 43	47 47
0x00003f0	29 29	33 33	45 45	49 49	19 19	21 21	23 23	25 25

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with channels 0 to 21 unused):

CLK start: 0x0000010

ULAP: 0x00000000

Used Channels:0x7fffffffffffc00000

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	30 30	32 32	34 34	36 36	38 38	40 40	42 42	22 22
0x0000030	24 24	26 26	28 28	30 30	32 32	34 34	36 36	38 38
0x0000050	40 40	42 42	44 44	46 46	48 48	50 50	52 52	54 54
0x0000070	56 56	58 58	60 60	62 62	32 32	36 36	34 34	38 38
0x0000090	40 40	44 44	42 42	46 46	48 48	52 52	50 50	54 54
0x00000b0	56 56	60 60	58 58	62 62	64 64	68 68	66 66	70 70
0x00000d0	72 72	76 76	74 74	78 78	45 45	49 49	47 47	51 51
0x00000f0	53 53	57 57	55 55	59 59	64 64	66 66	68 68	70 70
0x0000110	45 45	47 47	49 49	51 51	72 72	74 74	76 76	78 78
0x0000130	53 53	55 55	57 57	59 59	61 61	63 63	65 65	23 23
0x0000150	33 33	35 35	37 37	39 39	25 25	27 27	29 29	31 31
0x0000170	41 41	43 43	45 45	47 47	61 61	65 65	63 63	23 23
0x0000190	33 33	37 37	35 35	39 39	25 25	29 29	27 27	31 31
0x00001b0	41 41	45 45	43 43	47 47	49 49	53 53	51 51	55 55
0x00001d0	65 65	69 69	67 67	71 71	57 57	61 61	59 59	63 63
0x00001f0	73 73	77 77	75 75	66 66	49 49	51 51	57 57	59 59
0x0000210	53 53	55 55	61 61	63 63	65 65	67 67	73 73	75 75
0x0000230	69 69	71 71	77 77	66 66	68 68	70 70	76 76	78 78
0x0000250	72 72	74 74	23 23	25 25	27 27	29 29	26 26	28 28
0x0000270	22 22	24 24	30 30	32 32	68 68	72 72	76 76	23 23
0x0000290	70 70	74 74	78 78	25 25	27 27	22 22	26 26	30 30
0x00002b0	29 29	24 24	28 28	32 32	34 34	38 38	42 42	46 46
0x00002d0	36 36	40 40	44 44	48 48	50 50	54 54	58 58	62 62
0x00002f0	52 52	56 56	60 60	64 64	34 34	36 36	50 50	52 52
0x0000310	38 38	40 40	54 54	56 56	42 42	44 44	58 58	60 60
0x0000330	46 46	48 48	62 62	64 64	66 66	68 68	34 34	36 36
0x0000350	70 70	72 72	38 38	40 40	74 74	76 76	42 42	44 44
0x0000370	78 78	32 32	46 46	48 48	66 66	70 70	34 34	38 38
0x0000390	68 68	72 72	36 36	40 40	74 74	78 78	42 42	46 46
0x00003b0	76 76	32 32	44 44	48 48	50 50	23 23	35 35	39 39
0x00003d0	52 52	25 25	37 37	41 41	27 27	31 31	43 43	47 47
0x00003f0	29 29	33 33	45 45	49 49	50 50	52 52	23 23	25 25

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with even channels used):

CLK start: 0x0000010

ULAP: 0x00000000

Used Channels:0x55555555555555555555

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	08 08	10 10	12 12	14 14	16 16	18 18	20 20	22 22
0x0000030	24 24	26 26	28 28	30 30	32 32	34 34	36 36	38 38
0x0000050	40 40	42 42	44 44	46 46	48 48	50 50	52 52	54 54
0x0000070	56 56	58 58	60 60	62 62	32 32	36 36	34 34	38 38
0x0000090	40 40	44 44	42 42	46 46	48 48	52 52	50 50	54 54
0x00000b0	56 56	60 60	58 58	62 62	64 64	68 68	66 66	70 70
0x00000d0	72 72	76 76	74 74	78 78	00 00	04 04	02 02	06 06
0x00000f0	08 08	12 12	10 10	14 14	64 64	66 66	68 68	70 70
0x0000110	00 00	02 02	04 04	06 06	72 72	74 74	76 76	78 78
0x0000130	08 08	10 10	12 12	14 14	16 16	18 18	20 20	22 22
0x0000150	32 32	34 34	36 36	38 38	24 24	26 26	28 28	30 30
0x0000170	40 40	42 42	44 44	46 46	16 16	20 20	18 18	22 22
0x0000190	32 32	36 36	34 34	38 38	24 24	28 28	26 26	30 30
0x00001b0	40 40	44 44	42 42	46 46	48 48	52 52	50 50	54 54
0x00001d0	64 64	68 68	66 66	70 70	56 56	60 60	58 58	62 62
0x00001f0	72 72	76 76	74 74	00 00	48 48	50 50	56 56	58 58
0x0000210	52 52	54 54	60 60	62 62	64 64	66 66	72 72	74 74
0x0000230	68 68	70 70	76 76	00 00	02 02	04 04	10 10	12 12
0x0000250	06 06	08 08	14 14	16 16	18 18	20 20	26 26	28 28
0x0000270	22 22	24 24	30 30	32 32	02 02	06 06	10 10	14 14
0x0000290	04 04	08 08	12 12	16 16	18 18	22 22	26 26	30 30
0x00002b0	20 20	24 24	28 28	32 32	34 34	38 38	42 42	46 46
0x00002d0	36 36	40 40	44 44	48 48	50 50	54 54	58 58	62 62
0x00002f0	52 52	56 56	60 60	64 64	34 34	36 36	50 50	52 52
0x0000310	38 38	40 40	54 54	56 56	42 42	44 44	58 58	60 60
0x0000330	46 46	48 48	62 62	64 64	66 66	68 68	00 00	02 02
0x0000350	70 70	72 72	04 04	06 06	74 74	76 76	08 08	10 10
0x0000370	78 78	78 78	12 12	14 14	66 66	70 70	00 00	04 04
0x0000390	68 68	72 72	02 02	06 06	74 74	78 78	08 08	12 12
0x00003b0	76 76	78 78	10 10	14 14	16 16	20 20	32 32	36 36
0x00003d0	18 18	22 22	34 34	38 38	24 24	28 28	40 40	44 44
0x00003f0	26 26	30 30	42 42	46 46	16 16	18 18	20 20	22 22

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with odd channels used):

CLK start: 0x0000010

ULAP: 0x00000000

Used Channels:0x2aaaaaaaaaaaaaaaaaaa

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	09 09	11 11	13 13	15 15	17 17	19 19	21 21	23 23
0x0000030	25 25	27 27	29 29	31 31	33 33	35 35	37 37	39 39
0x0000050	41 41	43 43	45 45	47 47	49 49	51 51	53 53	55 55
0x0000070	57 57	59 59	61 61	63 63	33 33	37 37	35 35	39 39
0x0000090	41 41	45 45	43 43	47 47	49 49	53 53	51 51	55 55
0x00000b0	57 57	61 61	59 59	63 63	65 65	69 69	67 67	71 71
0x00000d0	73 73	77 77	75 75	01 01	01 01	05 05	03 03	07 07
0x00000f0	09 09	13 13	11 11	15 15	65 65	67 67	69 69	71 71
0x0000110	01 01	03 03	05 05	07 07	73 73	75 75	77 77	01 01
0x0000130	09 09	11 11	13 13	15 15	17 17	19 19	21 21	23 23
0x0000150	33 33	35 35	37 37	39 39	25 25	27 27	29 29	31 31
0x0000170	41 41	43 43	45 45	47 47	17 17	21 21	19 19	23 23
0x0000190	33 33	37 37	35 35	39 39	25 25	29 29	27 27	31 31
0x00001b0	41 41	45 45	43 43	47 47	49 49	53 53	51 51	55 55
0x00001d0	65 65	69 69	67 67	71 71	57 57	61 61	59 59	63 63
0x00001f0	73 73	77 77	75 75	03 03	49 49	51 51	57 57	59 59
0x0000210	53 53	55 55	61 61	63 63	65 65	67 67	73 73	75 75
0x0000230	69 69	71 71	77 77	03 03	05 05	07 07	13 13	15 15
0x0000250	09 09	11 11	17 17	19 19	21 21	23 23	29 29	31 31
0x0000270	25 25	27 27	33 33	35 35	05 05	09 09	13 13	17 17
0x0000290	07 07	11 11	15 15	19 19	21 21	25 25	29 29	33 33
0x00002b0	23 23	27 27	31 31	35 35	37 37	41 41	45 45	49 49
0x00002d0	39 39	43 43	47 47	51 51	53 53	57 57	61 61	65 65
0x00002f0	55 55	59 59	63 63	67 67	37 37	39 39	53 53	55 55
0x0000310	41 41	43 43	57 57	59 59	45 45	47 47	61 61	63 63
0x0000330	49 49	51 51	65 65	67 67	69 69	71 71	03 03	05 05
0x0000350	73 73	75 75	07 07	09 09	77 77	01 01	11 11	13 13
0x0000370	03 03	01 01	15 15	17 17	69 69	73 73	03 03	07 07
0x0000390	71 71	75 75	05 05	09 09	77 77	03 03	11 11	15 15
0x00003b0	01 01	01 01	13 13	17 17	19 19	23 23	35 35	39 39
0x00003d0	21 21	25 25	37 37	41 41	27 27	31 31	43 43	47 47
0x00003f0	29 29	33 33	45 45	49 49	19 19	21 21	23 23	25 25

2.2 SECOND SET

```
Hop sequence {k} for PAGE SCAN/INQUIRY SCAN SUBSTATE:
CLKN start:     0x0000000
ULAP:           0x2a96ef25
#ticks:         0000 | 1000 | 2000 | 3000 | 4000 | 5000 | 6000 | 7000 |
                -----------------------------------------------------
0x0000000:        49 |   13 |   17 |   51 |   55 |   19 |   23 |   53 |
0x0008000:        57 |   21 |   25 |   27 |   31 |   74 |   78 |   29 |
0x0010000:        33 |   76 |    1 |   35 |   39 |    3 |    7 |   37 |
0x0018000:        41 |    5 |    9 |   43 |   47 |   11 |   15 |   45 |
0x0020000:        49 |   13 |   17 |   51 |   55 |   19 |   23 |   53 |
0x0028000:        57 |   21 |   25 |   27 |   31 |   74 |   78 |   29 |
0x0030000:        33 |   76 |    1 |   35 |   39 |    3 |    7 |   37 |
0x0038000:        41 |    5 |    9 |   43 |   47 |   11 |   15 |   45 |

Hop sequence {k} for PAGE STATE/INQUIRY SUBSTATE:
CLKE start:     0x0000000
ULAP:           0x2a96ef25
#ticks:         00 01 02 03 | 04 05 06 07 | 08 09 0a 0b | 0c 0d 0e 0f |
                -----------------------------------------------------
0x0000000:      41 05 10 04 | 09 43 06 16 | 47 11 18 12 | 15 45 14 32 |
0x0000010:      49 13 34 28 | 17 51 30 24 | 55 19 26 20 | 23 53 22 40 |
0x0000020:      41 05 10 04 | 09 43 06 16 | 47 11 18 12 | 15 45 14 32 |
0x0000030:      49 13 34 28 | 17 51 30 24 | 55 19 26 20 | 23 53 22 40 |
...
0x0001000:      41 21 10 36 | 25 27 38 63 | 31 74 65 59 | 78 29 61 00 |
0x0001010:      33 76 02 75 | 01 35 77 71 | 39 03 73 67 | 07 37 69 08 |
0x0001020:      41 21 10 36 | 25 27 38 63 | 31 74 65 59 | 78 29 61 00 |
0x0001030:      33 76 02 75 | 01 35 77 71 | 39 03 73 67 | 07 37 69 08 |
...
0x0002000:      57 21 42 36 | 09 43 06 16 | 47 11 18 12 | 15 45 14 32 |
0x0002010:      49 13 34 28 | 17 51 30 24 | 55 19 26 20 | 23 53 22 40 |
0x0002020:      57 21 42 36 | 09 43 06 16 | 47 11 18 12 | 15 45 14 32 |
0x0002030:      49 13 34 28 | 17 51 30 24 | 55 19 26 20 | 23 53 22 40 |
...
0x0003000:      41 05 10 04 | 09 27 06 63 | 31 74 65 59 | 78 29 61 00 |
0x0003010:      33 76 02 75 | 01 35 77 71 | 39 03 73 67 | 07 37 69 08 |
0x0003020:      41 05 10 04 | 09 27 06 63 | 31 74 65 59 | 78 29 61 00 |
0x0003030:      33 76 02 75 | 01 35 77 71 | 39 03 73 67 | 07 37 69 08 |

Hop scquencc {k} for SLAVE PAGE RESPONSE SUBSTATE:
CLKN* =         0x0000010
ULAP:           0x2a96ef25
#ticks:         00 | 02 04 | 06 08 | 0a 0c | 0e 10 | 12 14 | 16 18 | 1a 1c | 1e
                -----------------------------------------------------------------
0x0000012:      34 | 13 28 | 17 30 | 51 24 | 55 26 | 19 20 | 23 22 | 53 40 | 57
0x0000032:      42 | 21 36 | 25 38 | 27 63 | 31 65 | 74 59 | 78 61 | 29 00 | 33
0x0000052:      02 | 76 75 | 01 77 | 35 71 | 39 73 | 03 67 | 07 69 | 37 08 | 41
0x0000072:      10 | 05 04 | 09 06 | 43 16 | 47 18 | 11 12 | 15 14 | 45 32 | 49

Hop sequence {k} for MASTER PAGE RESPONSE SUBSTATE:
Offset value:   24
CLKE* =         0x0000012
ULAP:           0x2a96ef25
#ticks:         00 02 | 04 06 | 08 0a | 0c 0e | 10 12 | 14 16 | 18 1a | 1c 1e |
                -----------------------------------------------------------------
0x0000014:      13 28 | 17 30 | 51 24 | 55 26 | 19 20 | 23 22 | 53 40 | 57 42 |
0x0000034:      21 36 | 25 38 | 27 63 | 31 65 | 74 59 | 78 61 | 29 00 | 33 02 |
0x0000054:      76 75 | 01 77 | 35 71 | 39 73 | 03 67 | 07 69 | 37 08 | 41 10 |
0x0000074:      05 04 | 09 06 | 43 16 | 47 18 | 11 12 | 15 14 | 45 32 | 49 34 |
```

Sample Data

Hop sequence {k} for CONNECTION STATE (Basic channel hopping sequence; ie, non-AFH):
CLK start: 0x0000010
ULAP: 0x2a96ef25

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010:	55 26	19 20	23 22	53 40	57 42	21 36	25 38	27 63
0x0000030:	31 65	74 59	78 61	29 00	33 02	76 75	01 77	35 71
0x0000050:	39 73	03 67	07 69	37 08	41 10	05 04	09 06	43 16
0x0000070:	47 18	11 12	15 14	45 32	02 66	47 60	49 64	04 54
0x0000090:	06 58	51 52	53 56	08 70	10 74	55 68	57 72	59 14
0x00000b0:	61 18	27 12	29 16	63 30	65 34	31 28	33 32	67 22
0x00000d0:	69 26	35 20	37 24	71 38	73 42	39 36	41 40	75 46
0x00000f0:	77 50	43 44	45 48	00 62	26 11	69 05	73 07	36 17
0x0000110:	40 19	04 13	08 15	38 25	42 27	06 21	10 23	12 48
0x0000130:	16 50	59 44	63 46	14 56	18 58	61 52	65 54	28 64
0x0000150:	32 66	75 60	00 62	30 72	34 74	77 68	02 70	20 01
0x0000170:	24 03	67 76	71 78	22 09	58 43	24 37	26 41	68 47
0x0000190:	70 51	36 45	38 49	72 55	74 59	40 53	42 57	44 78
0x00001b0:	46 03	12 76	14 01	48 07	50 11	16 05	18 09	60 15
0x00001d0:	62 19	28 13	30 17	64 23	66 27	32 21	34 25	52 31
0x00001f0:	54 35	20 29	22 33	56 39	19 04	62 63	66 00	07 73
0x0000210:	11 10	54 69	58 06	23 75	27 12	70 71	74 08	76 33
0x0000230:	01 49	44 29	48 45	13 35	17 51	60 31	64 47	05 41
0x0000250:	09 57	52 37	56 53	21 43	25 59	68 39	72 55	78 65
0x0000270:	03 02	46 61	50 77	15 67	51 36	17 18	19 34	41 24
0x0000290:	43 40	09 22	11 38	57 28	59 44	25 26	27 42	29 63
0x00002b0:	31 00	76 61	78 77	45 67	47 04	13 65	15 02	37 71
0x00002d0:	39 08	05 69	07 06	53 75	55 12	21 73	23 10	33 16
0x00002f0:	35 32	01 14	03 30	49 20	75 60	39 48	43 56	00 66
0x0000310:	04 74	47 62	51 70	08 68	12 76	55 64	59 72	61 18
0x0000330:	65 26	29 14	33 22	69 20	73 28	37 16	41 24	77 34
0x0000350:	02 42	45 30	49 38	06 36	10 44	53 32	57 40	63 50
0x0000370:	67 58	31 46	35 54	71 52	28 13	73 03	75 11	34 17
0x0000390:	36 25	02 15	04 23	42 21	44 29	10 19	12 27	14 48
0x00003b0:	16 56	61 46	63 54	22 52	24 60	69 50	71 58	30 64
0x00003d0:	32 72	77 62	00 70	38 68	40 76	06 66	08 74	18 01
0x00003f0:	20 09	65 78	67 07	26 05	44 29	32 23	36 25	70 43

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with all channel used; ie, AFH(79)):

CLK start: 0x0000010

ULAP: 0x2a96ef25

Used Channels:0x7fffffffffffffffffff

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	55 55	19 19	23 23	53 53	57 57	21 21	25 25	27 27
0x0000030	31 31	74 74	78 78	29 29	33 33	76 76	01 01	35 35
0x0000050	39 39	03 03	07 07	37 37	41 41	05 05	09 09	43 43
0x0000070	47 47	11 11	15 15	45 45	02 02	47 47	49 49	04 04
0x0000090	06 06	51 51	53 53	08 08	10 10	55 55	57 57	59 59
0x00000b0	61 61	27 27	29 29	63 63	65 65	31 31	33 33	67 67
0x00000d0	69 69	35 35	37 37	71 71	73 73	39 39	41 41	75 75
0x00000f0	77 77	43 43	45 45	00 00	26 26	69 69	73 73	36 36
0x0000110	40 40	04 04	08 08	38 38	42 42	06 06	10 10	12 12
0x0000130	16 16	59 59	63 63	14 14	18 18	61 61	65 65	28 28
0x0000150	32 32	75 75	00 00	30 30	34 34	77 77	02 02	20 20
0x0000170	24 24	67 67	71 71	22 22	58 58	24 24	26 26	68 68
0x0000190	70 70	36 36	38 38	72 72	74 74	40 40	42 42	44 44
0x00001b0	46 46	12 12	14 14	48 48	50 50	16 16	18 18	60 60
0x00001d0	62 62	28 28	30 30	64 64	66 66	32 32	34 34	52 52
0x00001f0	54 54	20 20	22 22	56 56	19 19	62 62	66 66	07 07
0x0000210	11 11	54 54	58 58	23 23	27 27	70 70	74 74	76 76
0x0000230	01 01	44 44	48 48	13 13	17 17	60 60	64 64	05 05
0x0000250	09 09	52 52	56 56	21 21	25 25	68 68	72 72	78 78
0x0000270	03 03	46 46	50 50	15 15	51 51	17 17	19 19	41 41
0x0000290	43 43	09 09	11 11	57 57	59 59	25 25	27 27	29 29
0x00002b0	31 31	76 76	78 78	45 45	47 47	13 13	15 15	37 37
0x00002d0	39 39	05 05	07 07	53 53	55 55	21 21	23 23	33 33
0x00002f0	35 35	01 01	03 03	49 49	75 75	39 39	43 43	00 00
0x0000310	04 04	47 47	51 51	08 08	12 12	55 55	59 59	61 61
0x0000330	65 65	29 29	33 33	69 69	73 73	37 37	41 41	77 77
0x0000350	02 02	45 45	49 49	06 06	10 10	53 53	57 57	63 63
0x0000370	67 67	31 31	35 35	71 71	28 28	73 73	75 75	34 34
0x0000390	36 36	02 02	04 04	42 42	44 44	10 10	12 12	14 14
0x00003b0	16 16	61 61	63 63	22 22	24 24	69 69	71 71	30 30
0x00003d0	32 32	77 77	00 00	38 38	40 40	06 06	08 08	18 18
0x00003f0	20 20	65 65	67 67	26 26	44 44	32 32	36 36	70 70

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with channels 0 to 21 unused):

CLK start: 0x0000010

ULAP: 0x2a96ef25

Used Channels:0x7fffffffffffffc00000

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	55 55	50 50	23 23	53 53	57 57	52 52	25 25	27 27
0x0000030	31 31	74 74	78 78	29 29	33 33	76 76	32 32	35 35
0x0000050	39 39	34 34	38 38	37 37	41 41	36 36	40 40	43 43
0x0000070	47 47	42 42	46 46	45 45	55 55	47 47	49 49	57 57
0x0000090	59 59	51 51	53 53	61 61	63 63	55 55	57 57	59 59
0x00000b0	61 61	27 27	29 29	63 63	65 65	31 31	33 33	67 67
0x00000d0	69 69	35 35	37 37	71 71	73 73	39 39	41 41	75 75
0x00000f0	77 77	43 43	45 45	53 53	26 26	69 69	73 73	36 36
0x0000110	40 40	57 57	61 61	38 38	42 42	59 59	63 63	65 65
0x0000130	69 69	59 59	63 63	67 67	71 71	61 61	65 65	28 28
0x0000150	32 32	75 75	53 53	30 30	34 34	77 77	55 55	73 73
0x0000170	24 24	67 67	71 71	22 22	58 58	24 24	26 26	68 68
0x0000190	70 70	36 36	38 38	72 72	74 74	40 40	42 42	44 44
0x00001b0	46 46	65 65	67 67	48 48	50 50	69 69	71 71	60 60
0x00001d0	62 62	28 28	30 30	64 64	66 66	32 32	34 34	52 52
0x00001f0	54 54	73 73	22 22	56 56	37 37	62 62	66 66	25 25
0x0000210	29 29	54 54	58 58	23 23	27 27	70 70	74 74	76 76
0x0000230	76 76	44 44	48 48	31 31	35 35	60 60	64 64	23 23
0x0000250	27 27	52 52	56 56	39 39	25 25	68 68	72 72	78 78
0x0000270	78 78	46 46	50 50	33 33	51 51	35 35	37 37	41 41
0x0000290	43 43	27 27	29 29	57 57	59 59	25 25	27 27	29 29
0x00002b0	31 31	76 76	78 78	45 45	47 47	31 31	33 33	37 37
0x00002d0	39 39	23 23	25 25	53 53	55 55	39 39	23 23	33 33
0x00002f0	35 35	76 76	78 78	49 49	75 75	39 39	43 43	40 40
0x0000310	44 44	47 47	51 51	48 48	52 52	55 55	59 59	61 61
0x0000330	65 65	29 29	33 33	69 69	73 73	37 37	41 41	77 77
0x0000350	42 42	45 45	49 49	46 46	50 50	53 53	57 57	63 63
0x0000370	67 67	31 31	35 35	71 71	28 28	73 73	75 75	34 34
0x0000390	36 36	42 42	44 44	42 42	44 44	50 50	52 52	54 54
0x00003b0	56 56	61 61	63 63	22 22	24 24	69 69	71 71	30 30
0x00003d0	32 32	77 77	40 40	38 38	40 40	46 46	48 48	58 58
0x00003f0	60 60	65 65	67 67	26 26	44 44	32 32	36 36	70 70

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with even channels used):

CLK start: 0x0000010

ULAP: 0x2a96ef25

Used Channels:0x55555555555555555555

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	52 52	16 16	20 20	50 50	54 54	18 18	22 22	24 24
0x0000030	28 28	74 74	78 78	26 26	30 30	76 76	78 78	32 32
0x0000050	36 36	00 00	04 04	34 34	38 38	02 02	06 06	40 40
0x0000070	44 44	08 08	12 12	42 42	02 02	44 44	46 46	04 04
0x0000090	06 06	48 48	50 50	08 08	10 10	52 52	54 54	56 56
0x00000b0	58 58	24 24	26 26	60 60	62 62	28 28	30 30	64 64
0x00000d0	66 66	32 32	34 34	68 68	70 70	36 36	38 38	72 72
0x00000f0	74 74	40 40	42 42	00 00	26 26	66 66	70 70	36 36
0x0000110	40 40	04 04	08 08	38 38	42 42	06 06	10 10	12 12
0x0000130	16 16	56 56	60 60	14 14	18 18	58 58	62 62	28 28
0x0000150	32 32	72 72	00 00	30 30	34 34	74 74	02 02	20 20
0x0000170	24 24	64 64	68 68	22 22	58 58	24 24	26 26	68 68
0x0000190	70 70	36 36	38 38	72 72	74 74	40 40	42 42	44 44
0x00001b0	46 46	12 12	14 14	48 48	50 50	16 16	18 18	60 60
0x00001d0	62 62	28 28	30 30	64 64	66 66	32 32	34 34	52 52
0x00001f0	54 54	20 20	22 22	56 56	14 14	62 62	66 66	02 02
0x0000210	06 06	54 54	58 58	18 18	22 22	70 70	74 74	76 76
0x0000230	76 76	44 44	48 48	08 08	12 12	60 60	64 64	00 00
0x0000250	04 04	52 52	56 56	16 16	20 20	68 68	72 72	78 78
0x0000270	78 78	46 46	50 50	10 10	46 46	12 12	14 14	36 36
0x0000290	38 38	04 04	06 06	52 52	54 54	20 20	22 22	24 24
0x00002b0	26 26	76 76	78 78	40 40	42 42	08 08	10 10	32 32
0x00002d0	34 34	00 00	02 02	48 48	50 50	16 16	18 18	28 28
0x00002f0	30 30	76 76	78 78	44 44	70 70	34 34	38 38	00 00
0x0000310	04 04	42 42	46 46	08 08	12 12	50 50	54 54	56 56
0x0000330	60 60	24 24	28 28	64 64	68 68	32 32	36 36	72 72
0x0000350	02 02	40 40	44 44	06 06	10 10	48 48	52 52	58 58
0x0000370	62 62	26 26	30 30	66 66	28 28	68 68	70 70	34 34
0x0000390	36 36	02 02	04 04	42 42	44 44	10 10	12 12	14 14
0x00003b0	16 16	56 56	58 58	22 22	24 24	64 64	66 66	30 30
0x00003d0	32 32	72 72	00 00	38 38	40 40	06 06	08 08	18 18
0x00003f0	20 20	60 60	62 62	26 26	44 44	32 32	36 36	70 70

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with odd channels used):

CLK start: 0x0000010

ULAP: 0x2a96ef25

Used Channels:0x2aaaaaaaaaaaaaaaaaaa

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	55 55	19 19	23 23	53 53	57 57	21 21	25 25	27 27
0x0000030	31 31	77 77	03 03	29 29	33 33	01 01	01 01	35 35
0x0000050	39 39	03 03	07 07	37 37	41 41	05 05	09 09	43 43
0x0000070	47 47	11 11	15 15	45 45	07 07	47 47	49 49	09 09
0x0000090	11 11	51 51	53 53	13 13	15 15	55 55	57 57	59 59
0x00000b0	61 61	27 27	29 29	63 63	65 65	31 31	33 33	67 67
0x00000d0	69 69	35 35	37 37	71 71	73 73	39 39	41 41	75 75
0x00000f0	77 77	43 43	45 45	05 05	31 31	69 69	73 73	41 41
0x0000110	45 45	09 09	13 13	43 43	47 47	11 11	15 15	17 17
0x0000130	21 21	59 59	63 63	19 19	23 23	61 61	65 65	33 33
0x0000150	37 37	75 75	05 05	35 35	39 39	77 77	07 07	25 25
0x0000170	29 29	67 67	71 71	27 27	63 63	29 29	31 31	73 73
0x0000190	75 75	41 41	43 43	77 77	01 01	45 45	47 47	49 49
0x00001b0	51 51	17 17	19 19	53 53	55 55	21 21	23 23	65 65
0x00001d0	67 67	33 33	35 35	69 69	71 71	37 37	39 39	57 57
0x00001f0	59 59	25 25	27 27	61 61	19 19	67 67	71 71	07 07
0x0000210	11 11	59 59	63 63	23 23	27 27	75 75	01 01	03 03
0x0000230	01 01	49 49	53 53	13 13	17 17	65 65	69 69	05 05
0x0000250	09 09	57 57	61 61	21 21	25 25	73 73	77 77	05 05
0x0000270	03 03	51 51	55 55	15 15	51 51	17 17	19 19	41 41
0x0000290	43 43	09 09	11 11	57 57	59 59	25 25	27 27	29 29
0x00002b0	31 31	03 03	05 05	45 45	47 47	13 13	15 15	37 37
0x00002d0	39 39	05 05	07 07	53 53	55 55	21 21	23 23	33 33
0x00002f0	35 35	01 01	03 03	49 49	75 75	39 39	43 43	07 07
0x0000310	11 11	47 47	51 51	15 15	19 19	55 55	59 59	61 61
0x0000330	65 65	29 29	33 33	69 69	73 73	37 37	41 41	77 77
0x0000350	09 09	45 45	49 49	13 13	17 17	53 53	57 57	63 63
0x0000370	67 67	31 31	35 35	71 71	35 35	73 73	75 75	41 41
0x0000390	43 43	09 09	11 11	49 49	51 51	17 17	19 19	21 21
0x00003b0	23 23	61 61	63 63	29 29	31 31	69 69	71 71	37 37
0x00003d0	39 39	77 77	07 07	45 45	47 47	13 13	15 15	25 25
0x00003f0	27 27	65 65	67 67	33 33	51 51	39 39	43 43	77 77

2.3 THIRD SET

```
Hop sequence {k} for PAGE SCAN/INQUIRY SCAN SUBSTATE:
CLKN start:      0x0000000
ULAP:            0x6587cba9
#ticks:          0000 | 1000 | 2000 | 3000 | 4000 | 5000 | 6000 | 7000 |
                 -----------------------------------------------------
0x0000000:         16 |   65 |   67 |   18 |   20 |   53 |   55 |    6 |
0x0008000:          8 |   57 |   59 |   10 |   12 |   69 |   71 |   22 |
0x0010000:         24 |   73 |   75 |   26 |   28 |   45 |   47 |   77 |
0x0018000:          0 |   49 |   51 |    2 |    4 |   61 |   63 |   14 |
0x0020000:         16 |   65 |   67 |   18 |   20 |   53 |   55 |    6 |
0x0028000:          8 |   57 |   59 |   10 |   12 |   69 |   71 |   22 |
0x0030000:         24 |   73 |   75 |   26 |   28 |   45 |   47 |   77 |
0x0038000:          0 |   49 |   51 |    2 |    4 |   61 |   63 |   14 |

Hop sequence {k} for PAGE STATE/INQUIRY SUBSTATE:
CLKE start:      0x0000000
ULAP:            0x6587cba9
#ticks:          00 01 02 03 | 04 05 06 07 | 08 09 0a 0b | 0c 0d 0e 0f |
                 -----------------------------------------------------
0x0000000:       00 49 36 38 | 51 02 42 40 | 04 61 44 46 | 63 14 50 48 |
0x0000010:       16 65 52 54 | 67 18 58 56 | 20 53 60 62 | 55 06 66 64 |
0x0000020:       00 49 36 38 | 51 02 42 40 | 04 61 44 46 | 63 14 50 48 |
0x0000030:       16 65 52 54 | 67 18 58 56 | 20 53 60 62 | 55 06 66 64 |
...
0x0001000:       00 57 36 70 | 59 10 74 72 | 12 69 76 78 | 71 22 03 01 |
0x0001010:       24 73 05 07 | 75 26 11 09 | 28 45 13 30 | 47 77 34 32 |
0x0001020:       00 57 36 70 | 59 10 74 72 | 12 69 76 78 | 71 22 03 01 |
0x0001030:       24 73 05 07 | 75 26 11 09 | 28 45 13 30 | 47 77 34 32 |
...
0x0002000:       08 57 68 70 | 51 02 42 40 | 04 61 44 46 | 63 14 50 48 |
0x0002010:       16 65 52 54 | 67 18 58 56 | 20 53 60 62 | 55 06 66 64 |
0x0002020:       08 57 68 70 | 51 02 42 40 | 04 61 44 46 | 63 14 50 48 |
0x0002030:       16 65 52 54 | 67 18 58 56 | 20 53 60 62 | 55 06 66 64 |
...
0x0003000:       00 49 36 38 | 51 10 42 72 | 12 69 76 78 | 71 22 03 01 |
0x0003010:       24 73 05 07 | 75 26 11 09 | 28 45 13 30 | 47 77 34 32 |
0x0003020:       00 49 36 38 | 51 10 42 72 | 12 69 76 78 | 71 22 03 01 |
0x0003030:       24 73 05 07 | 75 26 11 09 | 28 45 13 30 | 47 77 34 32 |

Hop sequence {k} for SLAVE PAGE RESPONSE SUBSTATE:
CLKN* =          0x0000010
ULAP:            0x6587cba9
#ticks:          00 | 02 04 | 06 08 | 0a 0c | 0e 10 | 12 14 | 16 18 | 1a 1c | 1e
                 ---------------------------------------------------------------
0x0000012:       52 | 65 54 | 67 58 | 18 56 | 20 60 | 53 62 | 55 66 | 06 64 | 08
0x0000032:       68 | 57 70 | 59 74 | 10 72 | 12 76 | 69 78 | 71 03 | 22 01 | 24
0x0000052:       05 | 73 07 | 75 11 | 26 09 | 28 13 | 45 30 | 47 34 | 77 32 | 00
0x0000072:       36 | 49 38 | 51 42 | 02 40 | 04 44 | 61 46 | 63 50 | 14 48 | 16

Hop sequence {k} for MASTER PAGE RESPONSE SUBSTATE:
Offset value:    24
CLKE* =          0x0000012
ULAP:            0x6587cba9
#ticks:          00 02 | 04 06 | 08 0a | 0c 0e | 10 12 | 14 16 | 18 1a | 1c 1e |
                 -----------------------------------------------------------------
0x0000014:       65 54 | 67 58 | 18 56 | 20 60 | 53 62 | 55 66 | 06 64 | 08 68 |
0x0000034:       57 70 | 59 74 | 10 72 | 12 76 | 69 78 | 71 03 | 22 01 | 24 05 |
0x0000054:       73 07 | 75 11 | 26 09 | 28 13 | 45 30 | 47 34 | 77 32 | 00 36 |
0x0000074:       49 38 | 51 42 | 02 40 | 04 44 | 61 46 | 63 50 | 14 48 | 16 52 |
```

Sample Data

Hop sequence {k} for CONNECTION STATE (Basic channel hopping sequence; ie, non-AFH):
CLK start: 0x0000010
ULAP: 0x6587cba9

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010:	20 60	53 62	55 66	06 64	08 68	57 70	59 74	10 72
0x0000030:	12 76	69 78	71 03	22 01	24 05	73 07	75 11	26 09
0x0000050:	28 13	45 30	47 34	77 32	00 36	49 38	51 42	02 40
0x0000070:	04 44	61 46	63 50	14 48	50 05	16 07	20 09	48 11
0x0000090:	52 13	06 15	10 17	38 19	42 21	08 23	12 25	40 27
0x00000b0:	44 29	22 31	26 33	54 35	58 37	24 39	28 41	56 43
0x00000d0:	60 45	77 62	02 64	30 66	34 68	00 70	04 72	32 74
0x00000f0:	36 76	14 78	18 01	46 03	72 29	42 39	44 43	74 41
0x0000110:	76 45	46 47	48 51	78 49	01 53	50 63	52 67	03 65
0x0000130:	05 69	54 55	56 59	07 57	09 61	58 71	60 75	11 73
0x0000150:	13 77	30 15	32 19	62 17	64 21	34 31	36 35	66 33
0x0000170:	68 37	38 23	40 27	70 25	27 61	72 71	76 73	25 75
0x0000190:	29 77	78 00	03 02	31 04	35 06	01 16	05 18	33 20
0x00001b0:	37 22	07 08	11 10	39 12	43 14	09 24	13 26	41 28
0x00001d0:	45 30	62 47	66 49	15 51	19 53	64 63	68 65	17 67
0x00001f0:	21 69	70 55	74 57	23 59	53 22	35 12	37 28	67 14
0x0000210:	69 30	23 32	25 48	55 34	57 50	39 40	41 56	71 42
0x0000230:	73 58	27 36	29 52	59 38	61 54	43 44	45 60	75 46
0x0000250:	77 62	15 00	17 16	47 02	49 18	31 08	33 24	63 10
0x0000270:	65 26	19 04	21 20	51 06	06 54	65 42	69 58	18 46
0x0000290:	22 62	55 64	59 01	08 68	12 05	71 72	75 09	24 76
0x00002b0:	28 13	57 66	61 03	10 70	14 07	73 74	77 11	26 78
0x00002d0:	30 15	47 32	51 48	00 36	04 52	63 40	67 56	16 44
0x00002f0:	20 60	49 34	53 50	02 38	38 78	12 05	14 13	44 07
0x0000310:	46 15	16 17	18 25	48 19	50 27	24 33	26 41	56 35
0x0000330:	58 43	20 21	22 29	52 23	54 31	28 37	30 45	60 39
0x0000350:	62 47	00 64	02 72	32 66	34 74	08 01	10 09	40 03
0x0000370:	42 11	04 68	06 76	36 70	70 31	42 35	46 43	74 39
0x0000390:	78 47	48 49	52 57	01 53	05 61	56 65	60 73	09 69
0x00003b0:	13 77	50 51	54 59	03 55	07 63	58 67	62 75	11 71
0x00003d0:	15 00	32 17	36 25	64 21	68 29	40 33	44 41	72 37
0x00003f0:	76 45	34 19	38 27	66 23	11 71	05 18	07 22	13 20

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with all channel used; ie, AFH(79)):

CLK start: 0x0000010

ULAP: 0x6587cba9

Used Channels:0x7fffffffffffffffffff

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	20 20	53 53	55 55	06 06	08 08	57 57	59 59	10 10
0x0000030	12 12	69 69	71 71	22 22	24 24	73 73	75 75	26 26
0x0000050	28 28	45 45	47 47	77 77	00 00	49 49	51 51	02 02
0x0000070	04 04	61 61	63 63	14 14	50 50	16 16	20 20	48 48
0x0000090	52 52	06 06	10 10	38 38	42 42	08 08	12 12	40 40
0x00000b0	44 44	22 22	26 26	54 54	58 58	24 24	28 28	56 56
0x00000d0	60 60	77 77	02 02	30 30	34 34	00 00	04 04	32 32
0x00000f0	36 36	14 14	18 18	46 46	72 72	42 42	44 44	74 74
0x0000110	76 76	46 46	48 48	78 78	01 01	50 50	52 52	03 03
0x0000130	05 05	54 54	56 56	07 07	09 09	58 58	60 60	11 11
0x0000150	13 13	30 30	32 32	62 62	64 64	34 34	36 36	66 66
0x0000170	68 68	38 38	40 40	70 70	27 27	72 72	76 76	25 25
0x0000190	29 29	78 78	03 03	31 31	35 35	01 01	05 05	33 33
0x00001b0	37 37	07 07	11 11	39 39	43 43	09 09	13 13	41 41
0x00001d0	45 45	62 62	66 66	15 15	19 19	64 64	68 68	17 17
0x00001f0	21 21	70 70	74 74	23 23	53 53	35 35	37 37	67 67
0x0000210	69 69	23 23	25 25	55 55	57 57	39 39	41 41	71 71
0x0000230	73 73	27 27	29 29	59 59	61 61	43 43	45 45	75 75
0x0000250	77 77	15 15	17 17	47 47	49 49	31 31	33 33	63 63
0x0000270	65 65	19 19	21 21	51 51	06 06	65 65	69 69	18 18
0x0000290	22 22	55 55	59 59	08 08	12 12	71 71	75 75	24 24
0x00002b0	28 28	57 57	61 61	10 10	14 14	73 73	77 77	26 26
0x00002d0	30 30	47 47	51 51	00 00	04 04	63 63	67 67	16 16
0x00002f0	20 20	49 49	53 53	02 02	38 38	12 12	14 14	44 44
0x0000310	46 46	16 16	18 18	48 48	50 50	24 24	26 26	56 56
0x0000330	58 58	20 20	22 22	52 52	54 54	28 28	30 30	60 60
0x0000350	62 62	00 00	02 02	32 32	34 34	08 08	10 10	40 40
0x0000370	42 42	04 04	06 06	36 36	70 70	42 42	46 46	74 74
0x0000390	78 78	48 48	52 52	01 01	05 05	56 56	60 60	09 09
0x00003b0	13 13	50 50	54 54	03 03	07 07	58 58	62 62	11 11
0x00003d0	15 15	32 32	36 36	64 64	68 68	40 40	44 44	72 72
0x00003f0	76 76	34 34	38 38	66 66	11 11	05 05	07 07	13 13

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with channels 0 to 21 unused):

CLK start: 0x0000010

ULAP: 0x6587cba9

Used Channels:0x7fffffffffffc00000

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	29 29	53 53	55 55	72 72	74 74	57 57	59 59	76 76
0x0000030	78 78	69 69	71 71	22 22	24 24	73 73	75 75	26 26
0x0000050	28 28	45 45	47 47	77 77	66 66	49 49	51 51	68 68
0x0000070	70 70	61 61	63 63	23 23	50 50	25 25	29 29	48 48
0x0000090	52 52	72 72	76 76	38 38	42 42	74 74	78 78	40 40
0x00000b0	44 44	22 22	26 26	54 54	58 58	24 24	28 28	56 56
0x00000d0	60 60	77 77	68 68	30 30	34 34	66 66	70 70	32 32
0x00000f0	36 36	23 23	27 27	46 46	72 72	42 42	44 44	74 74
0x0000110	76 76	46 46	48 48	78 78	32 32	50 50	52 52	34 34
0x0000130	36 36	54 54	56 56	38 38	40 40	58 58	60 60	42 42
0x0000150	44 44	30 30	32 32	62 62	64 64	34 34	36 36	66 66
0x0000170	68 68	38 38	40 40	70 70	27 27	72 72	76 76	25 25
0x0000190	29 29	78 78	34 34	31 31	35 35	32 32	36 36	33 33
0x00001b0	37 37	38 38	42 42	39 39	43 43	40 40	44 44	41 41
0x00001d0	45 45	62 62	66 66	46 46	50 50	64 64	68 68	48 48
0x00001f0	52 52	70 70	74 74	23 23	53 53	35 35	37 37	67 67
0x0000210	69 69	23 23	25 25	55 55	57 57	39 39	41 41	71 71
0x0000230	73 73	27 27	29 29	59 59	61 61	43 43	45 45	75 75
0x0000250	77 77	46 46	48 48	47 47	49 49	31 31	33 33	63 63
0x0000270	65 65	50 50	52 52	51 51	59 59	65 65	69 69	71 71
0x0000290	22 22	55 55	59 59	61 61	65 65	71 71	75 75	24 24
0x00002b0	28 28	57 57	61 61	63 63	67 67	73 73	77 77	26 26
0x00002d0	30 30	47 47	51 51	53 53	57 57	63 63	67 67	69 69
0x00002f0	73 73	49 49	53 53	55 55	38 38	65 65	67 67	44 44
0x0000310	46 46	69 69	71 71	48 48	50 50	24 24	26 26	56 56
0x0000330	58 58	73 73	22 22	52 52	54 54	28 28	30 30	60 60
0x0000350	62 62	53 53	55 55	32 32	34 34	61 61	63 63	40 40
0x0000370	42 42	57 57	59 59	36 36	70 70	42 42	46 46	74 74
0x0000390	78 78	48 48	52 52	76 76	23 23	56 56	60 60	27 27
0x00003b0	31 31	50 50	54 54	78 78	25 25	58 58	62 62	29 29
0x00003d0	33 33	32 32	36 36	64 64	68 68	40 40	44 44	72 72
0x00003f0	76 76	34 34	38 38	66 66	29 29	23 23	25 25	31 31

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with even channels used):

CLK start: 0x0000010

ULAP: 0x6587cba9

Used Channels:0x55555555555555555555

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	20 20	52 52	54 54	06 06	08 08	56 56	58 58	10 10
0x0000030	12 12	68 68	70 70	22 22	24 24	72 72	74 74	26 26
0x0000050	28 28	44 44	46 46	76 76	00 00	48 48	50 50	02 02
0x0000070	04 04	60 60	62 62	14 14	50 50	16 16	20 20	48 48
0x0000090	52 52	06 06	10 10	38 38	42 42	08 08	12 12	40 40
0x00000b0	44 44	22 22	26 26	54 54	58 58	24 24	28 28	56 56
0x00000d0	60 60	76 76	02 02	30 30	34 34	00 00	04 04	32 32
0x00000f0	36 36	14 14	18 18	46 46	72 72	42 42	44 44	74 74
0x0000110	76 76	46 46	48 48	78 78	78 78	50 50	52 52	00 00
0x0000130	02 02	54 54	56 56	04 04	06 06	58 58	60 60	08 08
0x0000150	10 10	30 30	32 32	62 62	64 64	34 34	36 36	66 66
0x0000170	68 68	38 38	40 40	70 70	24 24	72 72	76 76	22 22
0x0000190	26 26	78 78	00 00	28 28	32 32	78 78	02 02	30 30
0x00001b0	34 34	04 04	08 08	36 36	40 40	06 06	10 10	38 38
0x00001d0	42 42	62 62	66 66	12 12	16 16	64 64	68 68	14 14
0x00001f0	18 18	70 70	74 74	20 20	50 50	32 32	34 34	64 64
0x0000210	66 66	20 20	22 22	52 52	54 54	36 36	38 38	68 68
0x0000230	70 70	24 24	26 26	56 56	58 58	40 40	42 42	72 72
0x0000250	74 74	12 12	14 14	44 44	46 46	28 28	30 30	60 60
0x0000270	62 62	16 16	18 18	48 48	06 06	62 62	66 66	18 18
0x0000290	22 22	52 52	56 56	08 08	12 12	68 68	72 72	24 24
0x00002b0	28 28	54 54	58 58	10 10	14 14	70 70	74 74	26 26
0x00002d0	30 30	44 44	48 48	00 00	04 04	60 60	64 64	16 16
0x00002f0	20 20	46 46	50 50	02 02	38 38	12 12	14 14	44 44
0x0000310	46 46	16 16	18 18	48 48	50 50	24 24	26 26	56 56
0x0000330	58 58	20 20	22 22	52 52	54 54	28 28	30 30	60 60
0x0000350	62 62	00 00	02 02	32 32	34 34	08 08	10 10	40 40
0x0000370	42 42	04 04	06 06	36 36	70 70	42 42	46 46	74 74
0x0000390	78 78	48 48	52 52	76 76	00 00	56 56	60 60	04 04
0x00003b0	08 08	50 50	54 54	78 78	02 02	58 58	62 62	06 06
0x00003d0	10 10	32 32	36 36	64 64	68 68	40 40	44 44	72 72
0x00003f0	76 76	34 34	38 38	66 66	06 06	00 00	02 02	08 08

Sample Data

Hop Sequence {k} for CONNECTION STATE (Adapted channel hopping sequence with odd channels used):

CLK start: 0x0000010

ULAP: 0x6587cba9

Used Channels:0x2aaaaaaaaaaaaaaaaaaa

#ticks:	00 02	04 06	08 0a	0c 0e	10 12	14 16	18 1a	1c 1e
0x0000010	23 23	53 53	55 55	09 09	11 11	57 57	59 59	13 13
0x0000030	15 15	69 69	71 71	25 25	27 27	73 73	75 75	29 29
0x0000050	31 31	45 45	47 47	77 77	03 03	49 49	51 51	05 05
0x0000070	07 07	61 61	63 63	17 17	53 53	19 19	23 23	51 51
0x0000090	55 55	09 09	13 13	41 41	45 45	11 11	15 15	43 43
0x00000b0	47 47	25 25	29 29	57 57	61 61	27 27	31 31	59 59
0x00000d0	63 63	77 77	05 05	33 33	37 37	03 03	07 07	35 35
0x00000f0	39 39	17 17	21 21	49 49	75 75	45 45	47 47	77 77
0x0000110	01 01	49 49	51 51	03 03	01 01	53 53	55 55	03 03
0x0000130	05 05	57 57	59 59	07 07	09 09	61 61	63 63	11 11
0x0000150	13 13	33 33	35 35	65 65	67 67	37 37	39 39	69 69
0x0000170	71 71	41 41	43 43	73 73	27 27	75 75	01 01	25 25
0x0000190	29 29	03 03	03 03	31 31	35 35	01 01	05 05	33 33
0x00001b0	37 37	07 07	11 11	39 39	43 43	09 09	13 13	41 41
0x00001d0	45 45	65 65	69 69	15 15	19 19	67 67	71 71	17 17
0x00001f0	21 21	73 73	77 77	23 23	53 53	35 35	37 37	67 67
0x0000210	69 69	23 23	25 25	55 55	57 57	39 39	41 41	71 71
0x0000230	73 73	27 27	29 29	59 59	61 61	43 43	45 45	75 75
0x0000250	77 77	15 15	17 17	47 47	49 49	31 31	33 33	63 63
0x0000270	65 65	19 19	21 21	51 51	11 11	65 65	69 69	23 23
0x0000290	27 27	55 55	59 59	13 13	17 17	71 71	75 75	29 29
0x00002b0	33 33	57 57	61 61	15 15	19 19	73 73	77 77	31 31
0x00002d0	35 35	47 47	51 51	05 05	09 09	63 63	67 67	21 21
0x00002f0	25 25	49 49	53 53	07 07	43 43	17 17	19 19	49 49
0x0000310	51 51	21 21	23 23	53 53	55 55	29 29	31 31	61 61
0x0000330	63 63	25 25	27 27	57 57	59 59	33 33	35 35	65 65
0x0000350	67 67	05 05	07 07	37 37	39 39	13 13	15 15	45 45
0x0000370	47 47	09 09	11 11	41 41	75 75	47 47	51 51	01 01
0x0000390	05 05	53 53	57 57	01 01	05 05	61 61	65 65	09 09
0x00003b0	13 13	55 55	59 59	03 03	07 07	63 63	67 67	11 11
0x00003d0	15 15	37 37	41 41	69 69	73 73	45 45	49 49	77 77
0x00003f0	03 03	39 39	43 43	71 71	11 11	05 05	07 07	13 13

3 ACCESS CODE SAMPLE DATA

Different access codes (GIAC, DIACs, others...)

LAP with LSB as rightmost bit.

	Bit transmit order on air		
	---------->		
LAP:	Preamble:	Sync word:	Trailer:
000000	5	7e7041e3 4000000d	5
ffffff	a	e758b522 7ffffff2	a
9c8b33	5	475c58cc 73345c72	a
9e8b34	5	28ed3c34 cb345e72	a
9e8b36	5	62337b64 1b345e72	a
9e8b39	a	c05747b9 e7345e72	a
9e8b3d	5	7084eab0 2f345e72	a
9e8b42	5	64c86d2b 90b45e72	a
9e8b48	a	e3c3725e 04b45e72	a
9e8b4f	a	8c7216a6 bcb45e72	a
9e8b57	a	b2f16c30 fab45e72	a
9e8b60	5	57bd3b22 c1b45e72	a
9e8b6a	a	d0b62457 55b45e72	a
9e8b75	a	81843a39 abb45e72	a
9e8b81	5	0ca96681 e0745e72	a
9e8b8e	a	aecd5a5c 1c745e72	a
9e8b9c	5	17453fbf ce745e72	a
9e8bab	a	f20968ad f5745e72	a
9e8bbb	5	015f4a1e f7745e72	a
9e8bcc	a	d8c695a0 0cf45e72	a
9e8bde	5	614ef043 def45e72	a
9e8bf1	a	ba81ddc7 a3f45e72	a
9e8c05	5	64a7dc4f 680c5e72	a
9e8c1a	5	3595c221 960c5e72	a
9e8c30	a	cb35cc0d 830c5e72	a
9e8c47	5	12ac13b3 788c5e72	a
9e8c5f	5	2c2f6925 3e8c5e72	a
9e8c78	5	3a351c84 078c5e72	a
9e8c92	5	7396d0f3 124c5e72	a
9e8cad	5	5b0fdfc4 6d4c5e72	a
9e8cc9	a	aea2eb38 e4cc5e72	a
9e8ce6	5	756dc6bc 99cc5e72	a
9e8d04	5	214cf934 882c5e72	a
9e8d23	5	37568c95 b12c5e72	a
9e8d43	5	72281560 f0ac5e72	a
9e8d64	5	643260c1 c9ac5e72	a
9e8d86	a	e044f493 986c5e72	a
9e8da9	5	3b8bd917 e56c5e72	a
9e8dcd	a	ce26edeb 6cec5e72	a
9e8df2	a	e6bfe2dc 13ec5e72	a
9e8e18	a	82dcde3d c61c5e72	a
9e8e3f	a	94c6ab9c ff1c5e72	a

Sample Data

9e8e67	a	969059a6 799c5e72	a
9e8e90	a	c4dfccef 425c5e72	a
9e8eba	5	3a7fc2c3 575c5e72	a
9e8ee5	5	57985401 69dc5e72	a
9e8f11	5	0ae2a363 623c5e72	a
9e8f3e	a	d12d8ee7 1f3c5e72	a
9e8f6c	5	547063a8 0dbc5e72	a
9e8f9b	5	063ff6e1 367c5e72	a
9e8fcb	a	c9bc5cfe f4fc5e72	a
9e8ffc	5	2cf00bec cffc5e72	a
9e902e	a	8ec5052f 5d025e72	a
9e9061	5	1074b15e 61825e72	a
9e9095	a	9d59ede6 2a425e72	a
9e90ca	a	f0be7b24 14c25e72	a
9e9100	5	10e10dd0 c0225e72	a
9e9137	a	f5ad5ac2 fb225e72	a
9e916f	a	f7fba8f8 7da25e72	a
9e91a8	5	2f490e5b c5625e72	a
9e91e2	a	94979982 91e25e72	a
9e921d	5	26cda478 2e125e72	a
9e9259	a	aacb81dd 26925e72	a
9e9296	a	bfac7f5b da525e72	a
9e92d4	a	c9a7b0a7 cad25e72	a
9e9313	a	c142bdde 32325e72	a
616cec	5	586a491f 0dcda18d	5
616ceb	5	37db2de7 b5cda18d	5
616ce9	5	7d056ab7 65cda18d	5
616ce6	a	df61566a 99cda18d	5
616ce2	5	6fb2fb63 51cda18d	5
616cdd	5	472bf454 2ecda18d	5
616cd7	a	c020eb21 bacda18d	5
616cd0	a	af918fd9 02cda18d	5
616cc8	a	9112f54f 44cda18d	5
616cbf	5	488b2af1 bf4da18d	5
616cb5	a	cf803584 2b4da18d	5
616caa	a	9eb22bea d54da18d	5
616c9e	a	a49cb509 9e4da18d	5
616c91	5	06f889d4 624da18d	5
616c83	a	bf70ec37 b04da18d	5
616c74	a	ed3f797e 8b8da18d	5
616c64	5	1e695bcd 898da18d	5
616c53	a	fb250cdf b28da18d	5
616c41	5	42ad693c 608da18d	5
616c2e	a	a5b7cc14 dd0da18d	5
616c1a	a	9f9952f7 960da18d	5
616c05	a	ceab4c99 680da18d	5
616bef	a	d403ddde fdf5a18d	5
616bd8	5	314f8acc c6f5a18d	5
616bc0	5	0fccf05a 80f5a18d	5
616ba7	5	25030d57 7975a18d	5
616b8d	a	dba3037b 6c75a18d	5
616b72	5	4439ce17 13b5a18d	5

Sample Data

616b56	a	8d417247 5ab5a18d	5
616b39	5	6a5bd76f e735a18d	5
616b1b	5	592e8166 b635a18d	5
616afc	5	28609d46 cfd5a18d	5
616adc	5	51cb8c1f 4ed5a18d	5
616abb	5	7b047112 b755a18d	5
616a99	5	4871271b e655a18d	5
616a76	5	24bdc8c4 9b95a18d	5
616a52	a	edc57494 d295a18d	5
616a2d	a	f989f30f 6d15a18d	5
616a07	5	0729fd23 7815a18d	5
6169e0	a	8bf0ba4f 81e5a18d	5
6169b8	a	89a64875 0765a18d	5
61698f	5	6cea1f67 3c65a18d	5
616965	5	2549d310 29a5a18d	5
61693a	5	48ae45d2 1725a18d	5
61690e	5	7280db31 5c25a18d	5
6168e1	a	ce1b9f34 61c5a18d	5
6168b3	5	4b46727b 7345a18d	5
616884	a	ae0a2569 4845a18d	5
616854	a	ea5fc581 4a85a18d	5
616823	5	33c61a3f b105a18d	5
6167f1	a	c49fb8c5 63f9a18d	5
6167be	5	5a2e0cb4 5f79a18d	5
61678a	5	60009257 1479a18d	5
616755	a	86314e62 eab9a18d	5
61671f	5	3defd9bb be39a18d	5
6166e8	a	bff7e728 c5d9a18d	5
6166b0	a	bda11512 4359a18d	5
616677	5	6513b3b1 fb99a18d	5
61663d	a	decd2468 af19a18d	5
616602	a	f6542b5f d019a18d	5
6165c6	a	dc44b49b d8e9a18d	5
616589	5	42f500ea e469a18d	5
61654b	a	bf2885e1 34a9a18d	5
61650c	a	ec4c69b5 4c29a18d	5

4 HEC AND PACKET HEADER SAMPLE DATA

This section contains examples of HECs computed for sample UAP and packet header contents (Data). The resulting 54 bit packet headers are shown in the rightmost column. Note that the UAP, Data and HEC values are in hexadecimal notation, while the header is in octal notation. The header is transmitted from left to right over the air.

UAP	Data	HEC	Header (octal)
00	123	e1	770007 0C7070 000777
47	123	06	770007 0C7007 700000
00	124	32	007007 0C7007 007700
47	124	d5	007007 0C7070 707077
00	125	5a	707007 0C7007 077070
47	125	bd	707007 0C7070 777707
00	126	e2	077007 0C7007 000777
47	126	05	077007 0C7070 700000
00	127	8a	777007 0C7007 070007
47	127	6d	777007 0C7070 770770
00	11b	9e	770770 0C7007 777007
47	11b	79	770770 0C7070 077770
00	11c	4d	007770 0C7070 770070
47	11c	aa	007770 0C7007 070707
00	11d	25	707770 0C7070 700700
47	11d	c2	707770 0C7007 000077
00	11e	9d	077770 0C7070 777007
47	11e	7a	077770 0C7007 077770
00	11f	f5	777770 0C7070 707777
47	11f	12	777770 0C7007 007000

5 CRC SAMPLE DATA

This section shows the CRC computed for a sample 10 byte payload and a UAP of 0x47.

Data:
---

data[0] = 0x4e
data[1] = 0x01
data[2] = 0x02
data[3] = 0x03
data[4] = 0x04
data[5] = 0x05
data[6] = 0x06
data[7] = 0x07
data[8] = 0x08
data[9] = 0x09

UAP = 0x47

==> CRC = 6d d2

Codeword (hexadecimal notation):

4e 01 02 03 04 05 06 07 08 09 6d d2

NB: Over the air each byte in the codeword
is sent with the LSB first.

6 COMPLETE SAMPLE PACKETS

6.1 EXAMPLE OF DH1 PACKET

```
Packet header: (MSB...LSB)
--------------------------
LT_ADDR = 011
TYPE = 0100 (DH1)
FLOW = 0
ARQN = 1
SEQN = 0

Payload: (MSB...LSB)
--------------------
payload length: 5 bytes
logical channel = 10 (UA/I, Start L2CAP message)
flow = 1
data byte 1 = 00000001
data byte 2 = 00000010
data byte 3 = 00000011
data byte 4 = 00000100
data byte 5 = 00000101

HEC and CRC initialization: (MSB...LSB)
---------------------------------------
uap = 01000111

NO WHITENING USED




AIR DATA (LSB...MSB)

Packet header (incl HEC):
-------------------------
111111000
000000111000
000111000
000111111000000000000000

Payload (incl payload header and CRC):
--------------------------------------------------------------
01110100
10000000
01000000
11000000
00100000
10100000
1110110000110110
```

6.2 EXAMPLE OF DM1 PACKET

```
Packet header: (MSB...LSB)
--------------------------
LT_ADDR = 011
TYPE = 0011 (DM1)
FLOW = 0
ARQN = 1
SEQN = 0

Payload: (MSB...LSB)
--------------------
payload length: 5 bytes
logical channel = 10 (UA/I, Start L2CAP message)
flow = 1
data byte 1 = 00000001
data byte 2 = 00000010
data byte 3 = 00000011
data byte 4 = 00000100
data byte 5 = 00000101

HEC and CRC initialization: (MSB...LSB)
---------------------------------------
uap = 01000111

NO WHITENING USED




AIR DATA (LSB...MSB)

Packet header (incl HEC):
-------------------------
111111000
111111000000
000111000
111000000111111111111000

Payload (incl payload header, FEC23, CRC and 6 padded zeros):
-------------------------------------------------------------
0111010010 11001
0000000100 01011
0000110000 11110
0000100000 00111
1010000011 01100
1011000011 00010
0110000000 10001
```

7 WHITENING SEQUENCE SAMPLE DATA

This section shows the output of the whitening sequence generator.

Whitening Sequence (=D7)	Whitening LFSR D7.....D0
1	1111111
1	1101111
1	1001111
0	0001111
0	0011110
0	0111100
1	1111000
1	1100001
1	1010011
0	0110111
1	1101110
1	1001101
0	0001011
0	0010110
0	0101100
1	1011000
0	0100001
1	1000010
0	0010101
0	0101010
1	1010100
0	0111001
1	1110010
1	1110101
1	1111011
1	1100111
1	1011111
0	0101111
1	1011110
0	0101101
1	1011010
0	0100101
1	1001010
0	0000101
0	0001010
0	0010100
0	0101000
1	1010000
0	0110001
1	1100010
1	1010101
0	0111011
1	1110110

Sample Data

1	1111101
1	1101011
1	1000111
0	0011111
0	0111110
1	1111100
1	1101001
1	1000011
0	0010111
0	0101110
1	1011100
0	0101001
1	1010010
0	0110101
1	1101010
1	1000101
0	0011011
0	0110110
1	1101100
1	1001001
0	0000011
0	0000110
0	0001100
0	0011000
0	0110000
1	1100000
1	1010001
0	0110011
1	1100110
1	1011101
0	0101011
1	1010110
0	0111101
1	1111010
1	1100101
1	1011011
0	0100111
1	1001110
0	0001101
0	0011010
0	0110100
1	1101000
1	1000001
0	0010011
0	0100110
1	1001100
0	0001001
0	0010010
0	0100100
1	1001000
0	0000001
0	0000010

Sample Data

0	0000100
0	0001000
0	0010000
0	0100000
1	1000000
0	0010001
0	0100010
1	1000100
0	0011001
0	0110010
1	1100100
1	1011001
0	0100011
1	1000110
0	0011101
0	0111010
1	1110100
1	1111001
1	1100011
1	1010111
0	0111111
1	1111110
1	1101101
1	1001011
0	0000111
0	0001110
0	0011100
0	0111000
1	1110000
1	1110001
1	1110011
1	1110111
1	1111111

8 FEC SAMPLE DATA

```
================================================
 Rate 2/3 FEC -- (15,10) Shortened Hamming Code
================================================

Data is in hexadecimal notation, the codewords are in binary notation. The codeword bits
are sent from left to right over the air interface. The space in the codeword indicates
the start of parity bits.

Data:           Codeword:

0x001       1000000000 11010
0x002       0100000000 01101
0x004       0010000000 11100
0x008       0001000000 01110
0x010       0000100000 00111
0x020       0000010000 11001
0x040       0000001000 10110
0x080       0000000100 01011
0x100       0000000010 11111
0x200       0000000001 10101
```

9 ENCRYPTION KEY SAMPLE DATA

Explanation:

For the sections 9.1 - 9.5, the hexadecimal sample data is written with the least significant byte at the leftmost position and the most significant byte at the rightmost position. Within each byte, the *least significant bit* (LSB) is at the rightmost position and the *most significant bit* (MSB) is at the leftmost position Thus, a line reading:

aco: 48afcdd4bd40fef76693b113

means aco[0]=0x48, ac[1]=0xaf, ..., aco[11]=0x13. The LSB of aco[11] is '1' and the MSB of aco[11] is '0'.

Key [i]: denotes the ith sub-key in Ar or A'r;
round r: denotes the input to the rth round;
added ->: denotes the input to round 3 in A'r after adding original input (of round 1).

9.1 FOUR TESTS OF E1

```
rand    :00000000000000000000000000000000
address :000000000000
key     :00000000000000000000000000000000
round  1:00000000000000000000000000000000
Key [ 1]:00000000000000000000000000000000
Key [ 2]:4697b1baa3b7100ac537b3c95a28ac64
round  2:78d19f9307d2476a523ec7a8a026042a
Key [ 3]:ecabaac66795580df89af66e66dc053d
Key [ 4]:8ac3d8896ae9364943bfebd4969b68a0
round  3:600265247668dda0e81c07bbb30ed503
Key [ 5]:5d57921fd5715cbb22c1be7bbc996394
Key [ 6]:2a61b8343219fdfb1740e6511d41448f
round  4:d7552ef7cc9dbde568d80c2215bc4277
Key [ 7]:dd0480dee731d67f01a2f739da6f23ca
Key [ 8]:3ad01cd1303e12a1cd0fe0a8af82592c
round  5:fb06bef32b52ab8f2a4f2b6ef7f6d0cd
Key [ 9]:7dadb2efc287ce75061302904f2e7233
Key [10]:c08dcfa981e2c4272f6c7a9f52e11538
round  6:b46b711ebb3cf69e847a75f0ab884bdd
Key [11]:fc2042c708e409555e8c147660ffdfd7
Key [12]:fa0b21001af9a6b9e89e624cd99150d2
round  7:c585f308ff19404294f06b292e978994
Key [13]:18b40784ea5ba4c80ecb48694b4e9c35
Key [14]:454d54e5253c0c4a8b3fcca7db6baef4
round  8:2665fadb13acf952bf74b4ab12264b9f
Key [15]:2df37c6d9db52674f29353b0f011ed83
Key [16]:b60316733b1e8e70bd861b477e2456f1
Key [17]:884697b1baa3b7100ac537b3c95a28ac
```

Sample Data

```
round  1:158ffe43352085e8a5ec7a88e1ff2ba8
Key [ 1]:e9e5dfc1b3a79583e9e5dfc1b3a79583
Key [ 2]:7595bf57e0632c59f435c16697d4c864
round  2:0b5cc75febcdf7827ca29ec0901b6b5b
Key [ 3]:e31b96afcc75d286ef0ae257cbbc05b7
Key [ 4]:0d2a27b471bc0108c6263aff9d9b3b6b
round  3:e4278526c8429211f7f2f0016220aef4
added ->:f1b68365fd6217f952de6a89831fd95c
Key [ 5]:98d1eb5773cf59d75d3b17b3bc37c191
Key [ 6]:fd2b79282408ddd4ea0aa7511133336f
round  4:d0304ad18337f86040145d27aa5c8153
Key [ 7]:331227756638a41d57b0f7e071ee2a98
Key [ 8]:aa0dd8cc68b406533d0f1d64aabacf20
round  5:84db909d213bb0172b8b6aaf71bf1472
Key [ 9]:669291b0752e63f806fce76f10e119c8
Key [10]:ef8bdd46be8ee0277e9b78adef1ec154
round  6:f835f52921e903dfa762f1df5abd7f95
Key [11]:f3902eb06dc409cfd78384624964bf51
Key [12]:7d72702b21f97984a721c99b0498239d
round  7:ae6c0b4bb09f25c6a5d9788a31b605d1
Key [13]:532e60bceaf902c52a06c2c283ecfa32
Key [14]:181715e5192efb2a64129668cf5d9dd4
round  8:744a6235b86cc0b853cc9f74f6b65311
Key [15]:83017c1434342d4290e961578790f451
Key [16]:2603532f365604646ff65803795ccce5
Key [17]:882f7c907b565ea58dae1c928a0dcf41
sres    :056c0fe6
aco     :48afcdd4bd40fef76693b113
----------------------------------------
rand    :bc3f30689647c8d7c5a03ca80a91eceb
address :7ca89b233c2d
key     :159dd9f43fc3d328efba0cd8a861fa57
round  1:bc3f30689647c8d7c5a03ca80a91eceb
Key [ 1]:159dd9f43fc3d328efba0cd8a861fa57
Key [ 2]:326558b3c15551899a97790e65ff669e
round  2:3e950edf197615638cc19c09f8fedc9b
Key [ 3]:62e879b65b9f53bbfbd020c624b1d682
Key [ 4]:73415f30bac8ab61f410adc9442992db
round  3:6a7640791cb536678936c5ecd4ae5a73
Key [ 5]:5093cfa1d31c1c48acd76df030ea3c31
Key [ 6]:0b4acc2b8f1f694fc7bd91f4a70f3009
round  4:fca2c022a577e2ffb2aa007589693ec7
Key [ 7]:2ca43fc817947804ecff148d50d6f6c6
Key [ 8]:3fcd73524b533e00b7f7825bea2040a4
round  5:e97f8ea4ed1a6f4a36ffc179dc6bb563
Key [ 9]:6c67bec76ae8c8cc4d289f69436d3506
Key [10]:95ed95ee8cb97e61d75848464bffb379
round  6:38b07261d7340d028749de1773a415c7
Key [11]:ff566c1fc6b9da9ac502514550f3e9d2
Key [12]:ab5ce3f5c887d0f49b87e0d380e12f47
round  7:58241f1aed7c1c3e047d724331a0b774
Key [13]:a2cab6f95eac7d655dbe84a6cd4c47f5
```

Sample Data

```
Key [14]:f5caff88af0af8c42a20b5bbd2c8b460
round  8:3d1aaeff53c0910de63b9788b13c490f
Key [15]:185099c1131cf97001e2f36fda415025
Key [16]:a0ebb82676bc75e8378b189eff3f6b1d
Key [17]:cf5b348aaee27ae332b4f1bfa10289a6
round  1:2e4b417b9a2a9cfd7d8417d9a6a556eb
Key [ 1]:fe78b835f26468ab069fd3991b086fda
Key [ 2]:095c5a51c6fa6d3ac1d57fa19aa382bd
round  2:b8bca81d6bb45af9d92beadd9300f5ed
Key [ 3]:1af866df817fd9f4ec00bc704192cffc
Key [ 4]:f4a8a059c1f575f076f5fbb24bf16590
round  3:351aa16dec2c3a4787080249ed323eae
added ->:1b65e2167656d6bafa8c19904bd79445
Key [ 5]:8c9d18d9356a9954d341b4286e88ea1f
Key [ 6]:5c958d370102c9881bf753e69c7da029
round  4:2ce8fef47dda6a5bee74372e33e478a2
Key [ 7]:7eb2985c3697429fbe0da334bb51f795
Key [ 8]:af900f4b63a1138e2874bfb7c628b7b8
round  5:572787f563e1643c1c862b7555637fb4
Key [ 9]:834c8588dd8f3d4f31117a488420d69b
Key [10]:bc2b9b81c15d9a80262f3f48e9045895
round  6:16b4968c5d02853c3a43aa4cdb5f26ac
Key [11]:f08608c9e39ad3147cba61327919c958
Key [12]:2d4131decf4fa3a959084714a9e85c11
round  7:10e4120c7cccef9dd4ba4e6da8571b01
Key [13]:c934fd319c4a2b5361fa8eef05ae9572
Key [14]:4904c17aa47868e40471007cde3a97c0
round  8:f9081772498fed41b6ffd72b71fcf6c6
Key [15]:ea5e28687e97fa3f833401c86e6053ef
Key [16]:1168f58252c4ecfccafbdb3af857b9f2
Key [17]:b3440f69ef951b78b5cbd6866275301b
sres    :8d5205c5
aco     :3ed75df4abd9af638d144e94
----------------------------------------
rand    :0891caee063f5da1809577ff94ccdcfb
address :c62f19f6ce98
key     :45298d06e46bac21421ddfbed94c032b
round  1:0891caee063f5da1809577ff94ccdcfb
Key [ 1]:45298d06e46bac21421ddfbed94c032b
Key [ 2]:8f03e1e1fe1c191cad35a897bc400597
round  2:1c6ca013480a685c1b28e0317f7167e1
Key [ 3]:4f2ce3a092dde854ef496c8126a69e8e
Key [ 4]:968caee2ac6d7008c07283daec67f2f2
round  3:06b4915f5fcc1fc551a52048f0af8a26
Key [ 5]:ab0d5c31f94259a6bf85ee2d22edf56c
Key [ 6]:dfb74855c0085ce73dc17b84bfd50a92
round  4:077a92b040acc86e6e0a877db197a167
Key [ 7]:8f888952662b3db00d4e904e7ea53b5d
Key [ 8]:5e18bfcc07799b0132db88cd6042f599
round  5:7204881fb300914825fdc863e8ceadf3
Key [ 9]:bfca91ad9bd3d1a06c582b1d5512dddf
Key [10]:a88bc477e3fa1d5a59b5e6cf793c7a41
```

Sample Data

```
round  6:27031131d86cea2d747deb4f756143aa
Key [11]:f3cfb8dac8aea2a6a8ef95af3a2a2767
Key [12]:77beb90670c5300b03aa2b2232d3d40c
round  7:fc8c13d49149b1ce8d86f96e44a00065
Key [13]:b578373650af36a06e19fe335d726d32
Key [14]:6bcee918c7d0d24dfdf42237fcf99d53
round  8:04ef5f5a7ddf846cda0a07782fc23866
Key [15]:399f158241eb3e079f45d7b96490e7ea
Key [16]:1bcfbe98ecde2add52aa63ea79fb917a
Key [17]:ee8bc03ec08722bc2b075492873374af
round  1:d989d7a40cde7032d17b52f8117b69d5
Key [ 1]:2ecc6cc797cc41a2ab02007f6af396ae
Key [ 2]:acfaef7609c12567d537ae1cf9dc2198
round  2:8e76eb9a29b2ad5eea790db97aee37c1
Key [ 3]:079c8ff9b73d428df879906a0b87a6c8
Key [ 4]:19f2710baf403a494193d201f3a8c439
round  3:346bb7c35b2539676375aafe3af69089
added ->:edf48e675703a955b2f0fc062b71f95c
Key [ 5]:d623a6498f915cb2c8002765247b2f5a
Key [ 6]:900109093319bc30108b3d9434a77a72
round  4:fafb6c1f3ebbd2477be2da49dd923f69
Key [ 7]:e28e2ee6e72e7f4e5b5c11f10d204228
Key [ 8]:8e455cd11f8b9073a2dfa5413c7a4bc5
round  5:7c72230df588060a3cf920f9b0a08f06
Key [ 9]:28afb26e2c7a64238c41cefc16c53e74
Key [10]:d08dcafc2096395ba0d2dddd0e471f4d
round  6:55991df991db26ff00073a12baa3031d
Key [11]:fcffdcc3ad8faae091a7055b934f87c1
Key [12]:f8df082d77060252c02d91e55bd6a7d6
round  7:70ec682ff864375f63701fa4f6be5377
Key [13]:bef3706e523d708e8a44147d7508bc35
Key [14]:3e98ab283ca2422d56a56cf8b06caeb3
round  8:172f12ec933da85504b4ea5c90f8f0ea
Key [15]:87ad9625d06645d22598dd5ef811ea2c
Key [16]:8bd3db0cc8168009e5da90877e13a36f
Key [17]:0e74631d813a8351ac7039b348c41b42
sres    :00507e5f
aco     :2a5f19fbf60907e69f39ca9f
----------------------------------------
rand    :0ecd61782b4128480c05dc45542b1b8c
address :f428f0e624b3
key     :35949a914225fabad91995d226de1d92
round  1:0ecd61782b4128480c05dc45542b1b8c
Key [ 1]:35949a914225fabad91995d226de1d92
Key [ 2]:ea6b3dcccc8ee5d88de349fa5010404f
round  2:8935e2e263fbc4b9302cabdfc06bce3e
Key [ 3]:920f3a0f2543ce535d4e7f25ad80648a
Key [ 4]:ad47227edf9c6874e80ba80ebb95d2c9
round  3:b4c8b878675f184a0c72f3dab51f8f05
Key [ 5]:81a941ca7202b5e884ae8fa493ecac3d
Key [ 6]:bcde1520bee3660e86ce2f0fb78b9157
round  4:77ced9f2fc42bdd5c6312b87fc2377c5
```

Sample Data

```
Key [ 7]:c8eee7423d7c6efa75ecec0d2cd969d3
Key [ 8]:910b3f838a02ed441fbe863a02b4a1d0
round  5:fe28e8056f3004d60bb207e628b39cf2
Key [ 9]:56c647c1e865eb078348962ae070972d
Key [10]:883965da77ca5812d8104e2b640aec0d
round  6:1f2ba92259d9e88101518f145a33840f
Key [11]:61d4cb7e4f8868a283327806a9bd8d4d
Key [12]:9f57de3a3ff310e21dc1e696ce060304
round  7:cc9b5d0218d29037e88475152ebebb2f
Key [13]:7aa1d8adc1aeed7127ef9a18f6eb2d8e
Key [14]:b4db9da3bf865912acd14904c7f7785d
round  8:b04d352bedc02682e4a7f59d7cda1dba
Key [15]:a13d7141ef1f6c7d867e3d175467381b
Key [16]:08b2bc058e50d6141cdd566a307e1acc
Key [17]:057b2b4b4be5dc0ac49e50489b8006c9
round  1:5cfacc773bae995cd7f1b81e7c9ec7df
Key [ 1]:1e717950f5828f3930fe4a9395858815
Key [ 2]:d1623369b733d98bbc894f75866c544c
round  2:d571ffa21d9daa797b1a0a3c962fc64c
Key [ 3]:4abf27664ae364cc8a7e5bcf88214cc4
Key [ 4]:2aaedda8dc4933dd6aeaf6e5c0d5a482
round  3:e17c8e498a00f125bf654c938c23f36d
added ->:bd765a3eb1ae8a796856048df0c1bab2
Key [ 5]:bc7f8ab2d86000f47b1946cc8d7a7a2b
Key [ 6]:6b28544cb13ec6c5d98470df2cf900b7
round  4:a9727c26f2f06bd9920e83c8605dcd76
Key [ 7]:1be840d9107f2c9523f66bb19f5464a1
Key [ 8]:61d6fb1aa2f0c2b26fb2a3d6de8c177c
round  5:aeff751f146eab7e4626b2e2c9e2fb39
Key [ 9]:adabfc82570c568a233173099f23f4c2
Key [10]:b7df6b55ad266c0f1ff7452101f59101
round  6:cf412b95f454d5185e67ca671892e5bd
Key [11]:8e04a7282a2950dcbaea28f300e22de3
Key [12]:21362c114433e29bda3e4d51f803b0cf
round  7:16165722fe4e07ef88f8056b17d89567
Key [13]:710c8fd5bb3cbb5f132a7061de518bd9
Key [14]:0791de7334f4c87285809343f3ead3bd
round  8:28854cd6ad4a3c572b15490d4b81bc3f
Key [15]:4f47f0e5629a674bfcd13770eb3a3bd9
Key [16]:58a6d9a16a284cc0aead2126c79608a1
Key [17]:a564082a0a98399f43f535fd5cefad34
sres    :80e5629c
aco     :a6fe4dcde3924611d3cc6ba1


========================================
```

9.2 FOUR TESTS OF E21

```
rand    :00000000000000000000000000000000
address :000000000000
round  1:00000000000000000000000000000000
Key [ 1]:00000000000000000000000000000006
Key [ 2]:4697b1baa3b7100ac537b3c95a28dc94
round  2:98611307ab76bbde9a86af1ce8cad412
Key [ 3]:ecabaac66795580df89af66e665d863d
Key [ 4]:8ac3d8896ae9364943bfebd4a2a768a0
round  3:820999ad2e6618f4b578974beeedf9e7
added ->:820999ad2e6618f4b578974beeedf9e7
Key [ 5]:5d57921fd5715cbb22c1bedb1c996394
Key [ 6]:2a61b8343219fdfb1740e9541d41448f
round  4:acd6edec87581ac22dbdc64ea4ced3a2
Key [ 7]:dd0480dee731d67f01ba0f39da6f23ca
Key [ 8]:3ad01cd1303e12a18dcfe0a8af82592c
round  5:1c7798732f09fbfe25795a4a2fbc93c2
Key [ 9]:7dadb2efc287ce7b0c1302904f2e7233
Key [10]:c08dcfa981e2f4572f6c7a9f52e11538
round  6:c05b88b56aa70e9c40c79bb81cd911bd
Key [11]:fc2042c708658a555e8c147660ffdfd7
Key [12]:fa0b21002605a6b9e89e624cd99150d2
round  7:abacc71b481c84c798d1bdf3d62f7e20
Key [13]:18b407e44a5ba4c80ecb48694b4e9c35
Key [14]:454d57e8253c0c4a8b3fcca7db6baef4
round  8:e8204e1183ae85cf19edb2c86215b700
Key [15]:2d0b946d9db52674f29353b0f011ed83
Key [16]:76c316733b1e8e70bd861b477e2456f1
Key [17]:8e4697b1baa3b7100ac537b3c95a28ac
Ka      :d14ca028545ec262cee700e39b5c39ee
----------------------------------------
rand    :2dd9a550343191304013b2d7e1189d09
address :cac4364303b6
round  1:cac4364303b6cac4364303b6cac43643
Key [ 1]:2dd9a550343191304013b2d7e1189d0f
Key [ 2]:14c4335b2c43910c5dcc71d81a14242b
round  2:e169f788aad45a9011f11db5270b1277
Key [ 3]:55bfb712cba168d1a48f6e74cd9f4388
Key [ 4]:2a2b3aacca695caef2821b0fb48cc253
round  3:540f9c76652e92c44987c617035037bf
added ->:9ed3d23566e45c007fcac9a1c9146dfc
Key [ 5]:a06aab22d9a287384042976b4b6b00ee
Key [ 6]:c229d054bb72e8eb230e6dcdb32d16b7
round  4:83659a41675f7171ea57909dc5a79ab4
Key [ 7]:23c4812ab1905ddf77dedaed4105649a
Key [ 8]:40d87e272a7a1554ae2e85e3638cdf52
round  5:0b9382d0ed4f2fccdbb69d0db7b130a4
Key [ 9]:bdc064c6a39f6b84fe40db359f62a3c4
Key [10]:58228db841ce3cee983aa721f36aa1b9
round  6:c6ebda0f8f489792f09c189568226c1f
Key [11]:a815bacd6fa747a0d4f52883ac63ebe7
```

Sample Data

```
Key [12]:a9ce513b38ea006c333ecaaefcf1d0f8
round  7:75a8aba07e69c9065bcd831c40115116
Key [13]:3635e074792d4122130e5b824e52cd60
Key [14]:511bdb61bb28de72a5d794bffbf407df
round  8:57a6e279dcb764cf7dd6a749dd60c735
Key [15]:a32f5f21044b6744b6d913b13cdb4c0a
Key [16]:9722bbaeef281496ef8c23a9d41e92f4
Key [17]:807370560ad7e8a13a054a65a03b4049
Ka      :e62f8bac609139b3999aedbc9d228042
----------------------------------------
rand    :dab3cffe9d5739d1b7bf4a667ae5ee24
address :02f8fd4cd661
round  1:02f8fd4cd66102f8fd4cd66102f8fd4c
Key [ 1]:dab3cffe9d5739d1b7bf4a667ae5ee22
Key [ 2]:e315a8a65d809ec7c289e69c899fbdcc
round  2:ef85ff081b8709405e19f3e275cec7dc
Key [ 3]:df6a119bb50945fc8a3394e7216448f3
Key [ 4]:87fe86fb0d58b5dd0fb3b6b1dab51d07
round  3:aa25c21bf577d92dd97381e3e9edcc54
added ->:a81dbf5723d8dbd524bf5782ebe5c918
Key [ 5]:36cc253c506c0021c91fac9d8c469e90
Key [ 6]:d5fda00f113e303809b7f7d78a1a2b0e
round  4:9e69ce9b53caec3990894d2baed41e0d
Key [ 7]:c14b5edc10cabf16bc2a2ba4a8ae1e40
Key [ 8]:74c6131afc8dce7e11b03b1ea8610c16
round  5:a5460fa8cedca48a14fd02209e01f02e
Key [ 9]:346cfc553c6cbc9713edb55f4dcbc96c
Key [10]:bddf027cb059d58f0509f8963e9bdec6
round  6:92b33f11eadcacc5a43dd05f13d334dd
Key [11]:8eb9e040c36c4c0b4a7fd3dd354d53c4
Key [12]:c6ffecdd5e135b20879b9dfa4b34bf51
round  7:fb0541aa5e5df1a61c51aef606eb5a41
Key [13]:bf12f5a6ba08dfc4fda4bdfc68c997d9
Key [14]:37c4656b9215f3c959ea688fb64ad327
round  8:f0bbd2b94ae374346730581fc77a9c98
Key [15]:e87bb0d86bf421ea4f779a8eee3a866c
Key [16]:faa471e934fd415ae4c0113ec7f0a5ad
Key [17]:95204a80b8400e49db7cf6fd2fd40d9a
Ka      :b0376d0a9b338c2e133c32b69cb816b3
----------------------------------------
rand    :13ecad08ad63c37f8a54dc56e82f4dc1
address :9846c5ead4d9
round  1:9846c5ead4d99846c5ead4d99846c5ea
Key [ 1]:13ecad08ad63c37f8a54dc56e82f4dc7
Key [ 2]:ad04f127bed50b5e671d6510d392eaed
round  2:97374e18cdd0a6f7a5aa49d1ac875c84
Key [ 3]:57ad159e5774fa222f2f3039b9cd5101
Key [ 4]:9a1e9e1068fede02ef90496e25fd8e79
round  3:9dd3260373edd9d5f4e774826b88fd2d
added ->:0519ebe9a7c6719331d1485bf3cec2c7
Key [ 5]:378dce167db62920b0b392f7cfca316e
Key [ 6]:db4277795c87286faee6c9e9a6b71a93
```

Sample Data

```
round  4:40ec6563450299ac4e120d88672504d6
Key [ 7]:ec01aa2f5a8a793b36c1bb858d254380
Key [ 8]:2921a66cfa5bf74ac535424564830e98
round  5:57287bbb041bd6a56c2bd931ed410cd4
Key [ 9]:07018e45aab61b3c3726ee3d57dbd5f6
Key [10]:627381f0fa4c02b0c7d3e7dfbffc3333
round  6:66affa66a8dcd36e36bf6c3f1c6a276e
Key [11]:33b57c925bd5551999f716e138efbe79
Key [12]:a6dc7f9aa95bcc9243aebd12608f657a
round  7:450e65184fd8c72c578d5cdecb286743
Key [13]:a6a6db00fd8c72a28ea57ea542f6e102
Key [14]:dcf3377daeb2e24e61f0ad6620951c1f
round  8:e5eb180b519a4e673f21b7c4f4573f3d
Key [15]:621240b9506b462a7fa250da41844626
Key [16]:ae297810f01f43dc35756cd119ee73d6
Key [17]:b959835ec2501ad3894f8b8f1f4257f9
Ka      :5b61e83ad04d23e9d1c698851fa30447
========================================
```

9.3 THREE TESTS OF E22

(for K_master and overlay generation)

```
rand    :001de169248850245a5f7cc7f0d6d633
PIN     :d5a51083a04a1971f18649ea8b79311a
round  1:001de169248850245a5f7cc7f0d6d623
Key [ 1]:d5a51083a04a1971f18649ea8b79311a
Key [ 2]:7317cdbff57f9b99f9810a2525b17cc7
round  2:5f05c143347b59acae3cb002db23830f
Key [ 3]:f08bd258adf1d4ae4a54d8ccb26220b2
Key [ 4]:91046cbb4ccc43db18d6dd36ca7313eb
round  3:c8f3e3300541a25b6ac5a80c3105f3c4
added ->:c810c45921c9f27f302424cbc1dbc9e7
Key [ 5]:67fb2336f4d9f069da58d11c82f6bd95
Key [ 6]:4fed702c75bd72c0d3d8f38707134c50
round  4:bd5e0c3a97fa55b91a3bbbf306ebb978
Key [ 7]:41c947f80cdc0464c50aa89070af314c
Key [ 8]:680eecfa8daf41c7109c9a5cb1f26d75
round  5:21c1a762c3cc33e75ce8976a73983087
Key [ 9]:6e33fbd94d00ff8f72e8a7a0d2cebc4c
Key [10]:f4d726054c6b948add99fabb5733ddc3
round  6:56d0df484345582f6b574a449ba155eb
Key [11]:4eda2425546a24cac790f49ef2453b53
Key [12]:cf2213624ed1510408a5a3e00b7333df
round  7:120cf9963fe9ff22993f7fdf9600d9b8
Key [13]:d04b1a25b0b8fec946d5ecfa626d04c9
Key [14]:01e5611b0f0e140bdb64585fd3ae5269
round  8:a6337400ad8cb47fefb91332f5cb2713
Key [15]:f15b2dc433f534f61bf718770a3698b1
Key [16]:f990d0273d8ea2b9e0b45917a781c720
Key [17]:f41b3cc13d4301297bb6bdfcb3e5a1dd
Ka      :539e4f2732e5ae2de1e0401f0813bd0d
```

Sample Data

```
----------------------------------------
rand    :67ed56bfcf99825f0c6b349369da30ab
PIN     :7885b515e84b1f082cc499976f1725ce
round  1:67ed56bfcf99825f0c6b349369da30bb
Key [ 1]:7885b515e84b1f082cc499976f1725ce
Key [ 2]:72445901fdaf506beb036f4412512248
round  2:6b160b66a1f6c26c1f3432f463ef5aa1
Key [ 3]:59f0e4982e97633e5e7fd133af8f2c5b
Key [ 4]:b4946ec77a41bf7c729d191e33d458ab
round  3:3f22046c964c3e5ca2a26ec9a76a9f67
added ->:580f5ad359e5c003ae0da25ace44cfdc
Key [ 5]:eb0b839f97bdf534183210678520bbef
Key [ 6]:cff0bc4a94e5c8b2a2d24d9f59031e19
round  4:87aa61fc0ff88e744c195249b9a33632
Key [ 7]:592430f14d8f93db95dd691af045776d
Key [ 8]:3b55b404222bf445a6a2ef5865247695
round  5:83dcf592a854226c4dcd94e1ecf1bc75
Key [ 9]:a9714b86319ef343a28b87456416bd52
Key [10]:e6598b24390b3a0bf2982747993b0d78
round  6:dee0d13a52e96bcf7c72045a21609fc6
Key [11]:62051d8c51973073bff959b032c6e1e2
Key [12]:29e94f4ab73296c453c833e217a1a85b
round  7:08488005761e6c7c4dbb203ae453fe3a
Key [13]:0e255970b3e2fc235f59fc5acb10e8ce
Key [14]:d0dfbb3361fee6d4ffe45babf1cd7abf
round  8:0d81e89bddde7a7065316c47574feb8f
Key [15]:c12eee4eb38b7a171f0f736003774b40
Key [16]:8f962523f1c0abd9a087a0dfb11643d3
Key [17]:24be1c66cf8b022f12f1fb4c60c93fd1
Ka      :04435771e03a9daceb8bb1a493ee9bd8
----------------------------------------
rand    :40a94509238664f244ff8e3d13b119d3
PIN     :1ce44839badde30396d03c4c36f23006
round  1:40a94509238664f244ff8e3d13b119c3
Key [ 1]:1ce44839badde30396d03c4c36f23006
Key [ 2]:6dd97a8f91d628be4b18157af1a9dcba
round  2:0eac5288057d9947a24eabc1744c4582
Key [ 3]:fef9583d5f55fd4107ad832a725db744
Key [ 4]:fc3893507016d7c1db2bd034a230a069
round  3:60b424f1082b0cc3bd61be7b4c0155f0
added ->:205d69f82bb17031f9604c465fb26e33
Key [ 5]:0834d04f3e7e1f7f85f0c1db685ab118
Key [ 6]:1852397f9a3723169058e9b62bb3682b
round  4:2c6b65a49d66af6566675afdd6fa7d7d
Key [ 7]:6c10da21d762ae4ac1ba22a96d9007b4
Key [ 8]:9aa23658b90470a78d686344b8a9b0e7
round  5:a2c537899665113a42f1ac24773bdc31
Key [ 9]:137dee3bf879fe7bd02fe6d888e84f16
Key [10]:466e315a1863f47d0f93bc6827cf3450
round  6:e26982980d79b21ed3e20f8c3e71ba96
Key [11]:0b33cf831465bb5c979e6224d7f79f7c
Key [12]:92770660268ede827810d707a0977d73
```

Sample Data

```
round  7:e7b063c4e2e3110b89b7e1631c762dd5
Key [13]:7be30ae4961cf24ca17625a77bb7a9f8
Key [14]:be65574a33ae30e6e82dbd2826d3cc1a
round  8:7a963e37b2c2e76b489cfe40a2cf00e5
Key [15]:ed0ba7dd30d60a5e69225f0a33011e5b
Key [16]:765c990f4445e52b39e6ed6105ad1c4f
Key [17]:52627bf9f35d94f30d5b07ef15901adc
Ka      :9cde4b60f9b5861ed9df80858bac6f7f


========================================
```

9.4 TESTS OF E22 WITH PIN AUGMENTING

for PIN lengths 1,...,16 bytes

```
rand    :24b101fd56117d42c0545a4247357048
PIN length =16 octets
PIN     :fd397c7f5c1f937cdf82d8816cc377e2
round  1:24b101fd56117d42c0545a4247357058
Key [ 1]:fd397c7f5c1f937cdf82d8816cc377e2
Key [ 2]:0f7aac9c9b53f308d9fdbf2c78e3c30e
round  2:838edfe1226266953ccba8379d873107
Key [ 3]:0b8ac18d4bb44fad2efa115e43945abc
Key [ 4]:887b16b062a83bfa469772c25b456312
round  3:8cd0c9283120aba89a7f9d635dd4fe3f
added ->:a881cad5673128ea5ad3f7211a096e67
Key [ 5]:2248cbe6d299e9d3e8fd35a91178f65b
Key [ 6]:b92af6237385bd31f8fb57fb1bdd824e
round  4:2648d9c618a622b10ef80c4dbaf68b99
Key [ 7]:2bf5ffe84a37878ede2d4c30be60203b
Key [ 8]:c9cb6cec60cb8a8f29b99fcf3e71e40f
round  5:b5a7d9e96f68b14ccebf361de3914d0f
Key [ 9]:5c2f8a702e4a45575b103b0cce8a91c6
Key [10]:d453db0c9f9ddbd11e355d9a34d9b11b
round  6:632a091e7eefe1336857ddafd1ff3265
Key [11]:32805db7e59c5ed4acabf38d27e3fece
Key [12]:fde3a8eedfa3a12be09c1a8a00890fd7
round  7:048531e9fd3efa95910540150f8b137b
Key [13]:def07eb23f3a378f059039a2124bc4c2
Key [14]:2608c58f23d84a09b9ce95e5caac1ab4
round  8:461814ec7439d412d0732f0a6f799a6a
Key [15]:0a7ed16481a623e56ee1442ffa74f334
Key [16]:12add59aca0d19532f1516979954e369
Key [17]:dd43d02d39ffd6a386a4b98b4ac6eb23
Ka      :a5f2adf328e4e6a2b42f19c8b74ba884
----------------------------------------
rand    :321964061ac49a436f9fb9824ac63f8b
PIN length =15 octets
PIN     :ad955d58b6b8857820ac1262d617a6
address :0314c0642543
round  1:321964061ac49a436f9fb9824ac63f9b
```

Sample Data

```
Key [ 1]:ad955d58b6b8857820ac1262d617a603
Key [ 2]:f281736f68e3d30b2ac7c67f125dc416
round  2:7c4a4ece1398681f4bafd309328b7770
Key [ 3]:43c157f4c8b360387c32ab330f9c9aa8
Key [ 4]:3a3049945a298f6d076c19219c47c3cb
round  3:9672b00738bdfaf9bd92a855bc6f3afb
added ->:a48b1401228194bad23161d7f6357960
Key [ 5]:c8e2eaa6d73b7de18f3228ab2173bc69
Key [ 6]:8623f44488222e66a293677cf30bf2bb
round  4:9b30247aad3bf133712d034b46d21c68
Key [ 7]:f3e500902fba31db9bae50ef30e484a4
Key [ 8]:49d4b1137c18f4752dd9955a5a8d2f43
round  5:4492c25fda08083a768b4b5588966b23
Key [ 9]:9d59c451989e74785cc097eda7e42ab8
Key [10]:251de25f3917dcd99c18646107a641fb
round  6:21ae346635714d2623041f269978c0ee
Key [11]:80b8f78cb1a49ec0c3e32a238e60fddf
Key [12]:beb84f4d20a501e4a24ecfbde481902b
round  7:9b56a3d0f8932f20c6a77a229514fb00
Key [13]:852571b44f35fd9d9336d3c1d2506656
Key [14]:d0a0d510fb06ba76e69b8ee3ebc1b725
round  8:6cd8492b2fd31a86978bcdf644eb08a8
Key [15]:c7ffd523f32a874ed4a93430a25976de
Key [16]:16cdcb25e62964876d951fdcc07030d3
Key [17]:def32c0e12596f9582e5e3c52b303f52
Ka      :c0ec1a5694e2b48d54297911e6c98b8f
----------------------------------------
rand    :d4ae20c80094547d7051931b5cc2a8d6
PIN length =14 octets
PIN     :e1232e2c5f3b833b3309088a87b6
address :fabecc58e609
round  1:d4ae20c80094547d7051931b5cc2a8c6
Key [ 1]:e1232e2c5f3b833b3309088a87b6fabe
Key [ 2]:5f0812b47cd3e9a30d7707050fffa1f2
round  2:1f45f16be89794bef33e4547c9c0916a
Key [ 3]:77b681944763244ffa3cd71b248b79b5
Key [ 4]:e2814e90e04f485958ce58c9133e2be6
round  3:b10d2f4ac941035263cee3552d774d2f
added ->:65bb4f82c9d5572f131f764e7139f5e9
Key [ 5]:520acad20801dc639a2c6d66d9b79576
Key [ 6]:c72255cdb61d42be72bd45390dd25ba5
round  4:ead4dc34207b6ea721c62166e155aaad
Key [ 7]:ebf04c02075bf459ec9c3ec06627d347
Key [ 8]:a1363dd2812ee800a4491c0c74074493
round  5:f507944f3018e20586d81d7f326aae9d
Key [ 9]:b0b6ba79493dc833d7f425be7b8dadb6
Key [10]:08cd23e536b9b9b53e85eb004cba3111
round  6:fff450f4302a2b3571e8405e148346da
Key [11]:fec22374c6937dcd26171f4d2edfada3
Key [12]:0f1a8ef5979c69ff44f620c2e007b6e4
round  7:de558779589897f3402a90ee78c3f921
Key [13]:901fb66f0779d6aad0c0fba1fe812cb5
```

Sample Data

```
Key [14]:a0cab3cd15cd23603adc8d4474efb239
round  8:b2df0aa0c9f07fbbaa02f510a29cf540
Key [15]:18edc3f4296dd6f1dea13f7c143117a1
Key [16]:8d3d52d700a379d72ded81687f7546c7
Key [17]:5927badfe602f29345f840bb53e1dea6
Ka      :d7b39be13e3692c65b4a9e17a9c55e17
----------------------------------------
rand    :272b73a2e40db52a6a61c6520549794a
PIN length =13 octets
PIN     :549f2694f353f5145772d8ae1e
address :20487681eb9f
round  1:272b73a2e40db52a6a61c6520549795a
Key [ 1]:549f2694f353f5145772d8ae1e204876
Key [ 2]:42c855593d66b0c458fd28b95b6a5fbf
round  2:d7276dc8073f7677c31f855bde9501e2
Key [ 3]:75d0a69ae49a2da92e457d767879df52
Key [ 4]:b3aa7e7492971afaa0fb2b64827110df
round  3:71aae503831133d19bc452da4d0e409b
added ->:56d558a1671ee8fbf12518884857b9c1
Key [ 5]:9c8cf1604a98e9a503c342e272de5cf6
Key [ 6]:d35bc2df6b85540a27642106471057d9
round  4:f41a709c89ea80481aa3d2b9b2a9f8ca
Key [ 7]:b454dda74aeb4eff227ba48a58077599
Key [ 8]:bcba6aec050116aa9b7c6a9b7314d796
round  5:20fdda20f4a26b1bd38eb7f355a7be87
Key [ 9]:d41f8a9de0a716eb7167a1b6e321c528
Key [10]:5353449982247782d168ab43f17bc4d8
round  6:a70e316997eeed49a5a9ef9ba5e913b5
Key [11]:32cbc9cf1a81e36a45153972347ce4ac
Key [12]:5747619006cf4ef834c749f2c4b9feb6
round  7:e66f2317a825f589f76b47b6aa6e73fb
Key [13]:f9b68beba0a09d2a570a7dc88cc3c3c2
Key [14]:55718f9a4f0b1f9484e8c6b186a41a4b
round  8:5f68f940440a9798e074776019804ada
Key [15]:4ecc29be1b4d78433f6aa30db974a7fb
Key [16]:8470a066ffb00cda7b08059599f919f5
Key [17]:f39a36d74e960a051e1ca98b777848f4
Ka      :9ac64309a37c25c3b4a584fc002a1618
----------------------------------------
rand    :7edb65f01a2f45a2bc9b24fb3390667e
PIN length =12 octets
PIN     :2e5a42797958557b23447ca8
address :04f0d2737f02
round  1:7edb65f01a2f45a2bc9b24fb3390666e
Key [ 1]:2e5a42797958557b23447ca804f0d273
Key [ 2]:18a97c856561eb23e71af8e9e1be4799
round  2:3436e12db8ffdc1265cb5a86da2fed0b
Key [ 3]:7c0908dcbc73201e17c4f7aa1ab8aec8
Key [ 4]:7cb58833602fbe4194c7cc797ce8c454
round  3:caed6af4226f67e4ad1914620803ef2a
added ->:b4c8cf04389eac4611b438993b935544
Key [ 5]:f4dce7d607b5234562d0ebb2267b08b8
```

Sample Data

```
Key [ 6]:560b75c5545751fd8fa99fa4346e654b
round  4:ee67c87d6f74bb75db98f68bff0192c1
Key [ 7]:32f10cefd8d3e6424c6f91f1437808af
Key [ 8]:a934a46045be30fb3be3a5f3f7b18837
round  5:792398dcbeb8d10bdb07ae3c819e943c
Key [ 9]:a0f12e97c677a0e8ac415cd2c8a7ca88
Key [10]:e27014c908785f5ca03e8c6a1da3bf13
round  6:e778b6e0c3e8e7edf90861c7916d97a8
Key [11]:1b4a4303bcc0b2e0f41c72d47654bd9f
Key [12]:4b1302a50046026d6c9054fc8387965a
round  7:1fafddc7efa5f04c1dec1869d3f2d9bb
Key [13]:58c334bb543d49eca562cdbe0280e0fc
Key [14]:bdb60d383c692d06476b76646c8dec48
round  8:3d7c326d074bd6aa222ea050f04a3c7f
Key [15]:78c0162506be0b5953e8403c01028f93
Key [16]:24d7dbbe834dbd7b67f57fcf0d39d60f
Key [17]:2e74f1f3331c0f6585e87b2f715e187e
Ka      :d3af4c81e3f482f062999dee7882a73b
----------------------------------------
rand    :26a92358294dce97b1d79ec32a67e81a
PIN length =11 octets
PIN     :05fbad03f52fa9324f7732
address :b9ac071f9d70
round  1:26a92358294dce97b1d79ec32a67e80a
Key [ 1]:05fbad03f52fa9324f7732b9ac071f9d
Key [ 2]:2504c9691c04a18480c8802e922098c0
round  2:0be20e3d76888e57b6bf77f97a8714fb
Key [ 3]:576b2791d1212bea8408212f2d43e77e
Key [ 4]:90ae36dcce8724adb618f912d1b27297
round  3:1969667060764453257d906b7e58bd5b
added ->:3f12892849c312c494542ea854bfa551
Key [ 5]:bc492c42c9e87f56ec31af5474e9226e
Key [ 6]:c135d1dbed32d9519acfb4169f3e1a10
round  4:ac404205118fe771e54aa6f392da1153
Key [ 7]:83ccbdbbaf17889b7d18254dc9252fa1
Key [ 8]:80b90a1767d3f2848080802764e21711
round  5:41795e89ae9a0cf776ffece76f47fd7a
Key [ 9]:cc24e4a86e8eed129118fd3d5223a1dc
Key [10]:7b1e9c0eb9dab083574be7b7015a62c9
round  6:29ca9e2f87ca00370ef1633505bfba4b
Key [11]:888e6d88cf4beb965cf7d4f32b696baa
Key [12]:6d642f3e5510b0b043a44daa2cf5eec0
round  7:81fc891c3c6fd99acc00028a387e2366
Key [13]:e224f85da2ab63a23e2a3a036e421358
Key [14]:c8dc22aaa739e2cb85d6a0c08226c7d0
round  8:e30b537e7a000e3d2424a9c0f04c4042
Key [15]:a969aa818c6b324bae391bedcdd9d335
Key [16]:6974b6f2f07e4c55f2cc0435c45bebd1
Key [17]:134b925ebd98e6b93c14aee582062fcb
Ka      :be87b44d079d45a08a71d15208c5cb50
----------------------------------------
rand    :0edef05327eab5262430f21fc91ce682
```

Sample Data

```
PIN length =10 octets
PIN     :8210e47390f3f48c32b3
address :7a3cdfe377d1
round  1:0edef05327eab5262430f21fc91ce692
Key [ 1]:8210e47390f3f48c32b37a3cdfe377d1
Key [ 2]:c6be4c3e425e749b620a94c779e33a7e
round  2:07ca3c7a7a6bcbc31d79a856d9cffc0e
Key [ 3]:2587cec2a4b8e4f996a9ed664350d5dd
Key [ 4]:70e4bf72834d9d3dbb7eb2c239216dc0
round  3:792ad2ac4e4559d1463714d2f161b6f4
added ->:7708c2ff692f0ef7626706cd387d9c66
Key [ 5]:6696e1e7f8ac037e1fff3598f0c164e2
Key [ 6]:23dbfe4d0b561bea08fbcef25e49b648
round  4:7d8c71a9d7fbdcbd851bdf074550b100
Key [ 7]:b03648acd021550edee904431a02f00c
Key [ 8]:cb169220b7398e8f077730aa4bf06baa
round  5:b6fcaa45064ffd557e4b7b30cfbb83e0
Key [ 9]:af602c2ba16a454649951274c2be6527
Key [10]:5d60b0a7a09d524143eca13ad680bc9c
round  6:b3416d391a0c26c558843debd0601e9e
Key [11]:9a2f39bfe558d9f562c5f09a5c3c0263
Key [12]:72cae8eebd7fabd9b1848333c2aab439
round  7:abe4b498d9c36ea97b8fd27d7f813913
Key [13]:15f27ea11e83a51645d487b81371d7dc
Key [14]:36083c8666447e03d33846edf444eb12
round  8:8032104338a945ba044d102eabda3b22
Key [15]:0a3a8977dd48f3b6c1668578befadd02
Key [16]:f06b6675d78ca0ee5b1761bdcdab516d
Key [17]:cbc8a7952d33aa0496f7ea2d05390b23
Ka      :bf0706d76ec3b11cce724b311bf71ff5
----------------------------------------
rand    :86290e2892f278ff6c3fb917b020576a
PIN length = 9 octets
PIN     :3dcdffcfd086802107
address :791a6a2c5cc3
round  1:86290e2892f278ff6c3fb917b0205765
Key [ 1]:3dcdffcfd086802107791a6a2c5cc33d
Key [ 2]:b4962f40d7bb19429007062a3c469521
round  2:1ec59ffd3065f19991872a7863b0ef02
Key [ 3]:eb9ede6787dd196b7e340185562bf28c
Key [ 4]:2964e58aacf7287d1717a35b100ae23b
round  3:f817406f1423fc2fe33e25152679eaaf
added ->:7e404e47861574d08f7dde02969941ca
Key [ 5]:6abf9a314508fd61e486fa4e376c3f93
Key [ 6]:6da148b7ee2632114521842cbb274376
round  4:e9c2a8fac22b8c7cf0c619e2b3f890ed
Key [ 7]:df889cc34fda86f01096d52d116e620d
Key [ 8]:5eb04b147dc39d1974058761ae7b73fc
round  5:444a8aac0efee1c02f8d38f8274b7b28
Key [ 9]:8426cc59eee391b2bd50cf8f1efef8b3
Key [10]:8b5d220a6300ade418da791dd8151941
round  6:9185f983db150b1bccab1e5c12eb63a1
```

Sample Data

```
Key [11]:82ba4ddef833f6a4d18b07aa011f2798
Key [12]:ce63d98794682054e73d0359dad35ec4
round  7:5eded2668f5916dfd036c09e87902886
Key [13]:da794357652e80c70ad8b0715dbe33d6
Key [14]:732ef2c0c3220b31f3820c375e27bb29
round  8:88a5291b4acbba009a85b7dd6a834b3b
Key [15]:3ce75a61d4b465b70c95d7ccd5799633
Key [16]:5df9bd2c3a17a840cdaafb76c171db7c
Key [17]:3f8364b089733d902bccb0cd3386846f
Ka      :cdb0cc68f6f6fbd70b46652de3ef3ffb
----------------------------------------
rand    :3ab52a65bb3b24a08eb6cd284b4b9d4b
PIN length = 8 octets
PIN     :d0fb9b6838d464d8
address :25a868db91ab
round  1:3ab52a65bb3b24a08eb6cd284b4b9d45
Key [ 1]:d0fb9b6838d464d825a868db91abd0fb
Key [ 2]:2573f47b49dad6330a7a9155b7ae8ba1
round  2:ad2ffdff408fcfab44941016a9199251
Key [ 3]:d2c5b8fb80cba13712905a589adaee71
Key [ 4]:5a3381511b338719fae242758dea0997
round  3:2ddc17e570d7931a2b1d13f6ace928a5
added ->:17914180cb12b7baa5d3e0dee734c5e0
Key [ 5]:e0a4d8ac27fbe2783b7bcb3a36a6224d
Key [ 6]:949324c6864deac3eca8e324853e11c3
round  4:62c1db5cf31590d331ec40ad692e8df5
Key [ 7]:6e67148088a01c2d4491957cc9ddc4aa
Key [ 8]:557431deab7087bb4c03fa27228f60c6
round  5:9c8933bc361f4bde4d1bda2b5f8bb235
Key [ 9]:a2551aca53329e70ade3fd2bb7664697
Key [10]:05d0ad35de68a364b54b56e2138738fe
round  6:9156db34136aa06655bf28a05be0596a
Key [11]:1616a6b13ce2f2895c722e8495181520
Key [12]:b12e78a1114847b01f6ed2f5a1429a23
round  7:84dcc292ed836c1c2d523f2a899a2ad5
Key [13]:316e144364686381944e95afd8a026bb
Key [14]:1ab551b88d39d97ea7a9fe136dbfe2e1
round  8:87bdcac878d777877f4eccf042cfee5e
Key [15]:70e21ab08c23c7544524b64492b25cc9
Key [16]:35f730f2ae2b950a49a1bf5c8b9f8866
Key [17]:2f16924c22db8b74e2eadf1ba4ebd37c
Ka      :983218718ca9aa97892e312d86dd9516
----------------------------------------
rand    :a6dc447ff08d4b366ff96e6cf207e179
PIN length = 7 octets
PIN     :9c57e10b4766cc
address :54ebd9328cb6
round  1:a6dc447ff08d4b366ff96e6cf207e174
Key [ 1]:9c57e10b4766cc54ebd9328cb69c57e1
Key [ 2]:00a609f4d61db26993c8177e3ee2bba8
round  2:1ed26b96a306d7014f4e5c9ee523b73d
Key [ 3]:646d7b5f9aaa528384bda3953b542764
```

Sample Data

```
Key [ 4]:a051a42212c0e9ad5c2c248259aca14e
round  3:a53f526db18e3d7d53edbfc9711041ed
added ->:031b9612411b884b3ce62da583172299
Key [ 5]:d1bd5e64930e7f838d8a33994462d8b2
Key [ 6]:5dc7e2291e32435665ebd6956bec3414
round  4:9438be308ec83f35c560e2796f4e0559
Key [ 7]:10552f45af63b0f15e2919ab37f64fe7
Key [ 8]:c44d5717c114a58b09207392ebe341f8
round  5:b79a7b14386066d339f799c40479cb3d
Key [ 9]:6886e47b782325568eaf59715a75d8ff
Key [10]:8e1e335e659cd36b132689f78c147bda
round  6:ef232462228aa166438d10c34e17424b
Key [11]:8843efeedd5c2b7c3304d647f932f4d1
Key [12]:13785aaedd0adf67abb4f01872392785
round  7:02d133fe40d15f1073673b36bba35abd
Key [13]:837d7ca2722419e6be3fae35900c3958
Key [14]:93f8442973e7fccf2e7232d1d057c73a
round  8:275506a3d08c84e94cc58ed60054505e
Key [15]:8a7a9edffa3c52918bc6a45f57d91f5d
Key [16]:f214a95d777f763c56109882c4b52c84
Key [17]:10e2ee92c5ea1ddc5eb010e55510c403
Ka      :9cd6650ead86323e87cafb1ff516d1e0
----------------------------------------
rand    :3348470a7ea6cc6eb81b40472133262c
PIN length = 6 octets
PIN     :fcad169d7295
address :430d572f8842
round  1:3348470a7ea6cc6eb81b404721332620
Key [ 1]:fcad169d7295430d572f8842fcad169d
Key [ 2]:b3479d4d4fd178c43e7bc5b0c7d8983c
round  2:af976da9225066d563e10ab955e6fc32
Key [ 3]:7112462b37d82dd81a2a35d9eb43cb7c
Key [ 4]:c5a7030f8497945ac7b84600d1d161fb
round  3:d08f826ebd55a0bd7591c19a89ed9bde
added ->:e3d7c964c3fb6cd3cdac01dda820c1fe
Key [ 5]:84b0c6ef4a63e4dff19b1f546d683df5
Key [ 6]:f4023edfc95d1e79ed4bb4de9b174f5d
round  4:6cd952785630dfc7cf81eea625e42c5c
Key [ 7]:ea38dd9a093ac9355918632c90c79993
Key [ 8]:dbba01e278ddc76380727f5d7135a7de
round  5:93573b29715154959782 64b88f330f7f
Key [ 9]:d4dc3a31be34e412210fafa6eca00776
Key [10]:39d1e190ee92b0ff16d92a8be58d2fa0
round  6:b3f01d5e7fe1ce6da7b46d8c389baf47
Key [11]:1eb081328d4bcf94c9117b12c5cf22ac
Key [12]:7e047c2c552f9f1414d946775fabfe30
round  7:0b833bff6106d5bae033b4ce5af5a924
Key [13]:e78e685d9b2a7e29e7f2a19d1bc38ebd
Key [14]:1b582272a3121718c4096d2d8602f215
round  8:23de0bbdc70850a7803f4f10c63b2c0f
Key [15]:8569e860530d9c3d48a0870dac33f676
Key [16]:6966b528fdd1dc222527052c8f6cf5a6
```

Sample Data

```
Key [17]:a34244c757154c53171c663b0b56d5c2
Ka      :98f1543ab4d87bd5ef5296fb5e3d3a21
----------------------------------------
rand    :0f5bb150b4371ae4e5785293d22b7b0c
PIN length = 5 octets
PIN     :b10d068bca
address :b44775199f29
round  1:0f5bb150b4371ae4e5785293d22b7b07
Key [ 1]:b10d068bcab44775199f29b10d068bca
Key [ 2]:aec70d1048f1bbd2c18040318a8402ad
round  2:342d2b79d7fb7cd110379742b9842c79
Key [ 3]:6d8d5cf338f29ef4420639ef488e4fa9
Key [ 4]:a1584117541b759ba6d9f7eb2bedcbba
round  3:9407e8e3e810603921bf81cfda62770a
added ->:9b6299b35c477addc437d35c088df20d
Key [ 5]:09a20676666aeed6f22176274eb433f4
Key [ 6]:840472c001add5811a054be5f5c74754
round  4:9a3ba953225a7862c0a842ed3d0b2679
Key [ 7]:fad9e45c8bf70a972fcd9bff0e8751f5
Key [ 8]:e8f30ff666dfd212263416496ff3b2c2
round  5:2c573b6480852e875df34b28a5c44509
Key [ 9]:964cdba0cf8d593f2fc40f96daf8267a
Key [10]:bcd65c11b13e1a70bcd4aafba8864fe3
round  6:21b0cc49e880c5811d24dee0194e6e9e
Key [11]:468c8548ea9653c1a10df6288dd03c1d
Key [12]:5d252d17af4b09d3f4b5f7b5677b8211
round  7:e6d6bdcd63e1d37d9883543ba86392fd
Key [13]:e814bf307c767428c67793dda2df95c7
Key [14]:4812b979fdc20f0ff0996f61673a42cc
round  8:e3dde7ce6bd7d8a34599aa04d6a760ab
Key [15]:5b1e2033d1cd549fc4b028146eb5b3b7
Key [16]:0f284c14fb8fe706a5343e3aa35af7b1
Key [17]:b1f7a4b7456d6b577fded6dc7a672e37
Ka      :c55070b72bc982adb972ed05d1a74ddb
----------------------------------------
rand    :148662a4baa73cfadb55489159e476e1
PIN length = 4 octets
PIN     :fb20f177
address :a683bd0b1896
round  1:148662a4baa73cfadb55489159e476eb
Key [ 1]:fb20f177a683bd0b1896fb20f177a683
Key [ 2]:47266cefbfa468ca7916b458155dc825
round  2:3a942eb6271c3f4e433838a5d3fcbd27
Key [ 3]:688853a6d6575eb2f6a2724b0fbc133b
Key [ 4]:7810df048019634083a2d9219d0b5fe0
round  3:9c835b98a063701c0887943596780769
added ->:8809bd3c1a0aace6d3dcdca4cf5c7d82
Key [ 5]:c78f6dcf56da1bbd413828b33f5865b3
Key [ 6]:eb3f3d407d160df3d293a76d1a513c4a
round  4:7e68c4bafa020a4a59b5a1968105bab5
Key [ 7]:d330e038d6b19d5c9bb0d7285a360064
Key [ 8]:9bd3ee50347c00753d165faced702d9c
```

Sample Data

```
round  5:227bad0cf0838bdb15b3b3872c24f592
Key [ 9]:9543ad0fb3fe74f83e0e2281c6d4f5f0
Key [10]:746cd0383c17e0e80e6d095a87fd0290
round  6:e026e98c71121a0cb739ef6f59e14d26
Key [11]:fa28bea4b1c417536608f11f406ea1dd
Key [12]:3aee0f4d21699df9cb8caf5354a780ff
round  7:cd6a6d8137d55140046f8991da1fa40a
Key [13]:372b71bc6d1aa6e785358044fbcf05f4
Key [14]:00a01501224c0405de00aa2ce7b6ab04
round  8:52cd7257fe8d0c782c259bcb6c9f5942
Key [15]:c7015c5c1d7c030e00897f104a006d4a
Key [16]:260a9577790c62e074e71e19fd2894df
Key [17]:c041b7a231493acd15ddcdaee94b9f52
Ka      :7ec864df2f1637c7e81f2319ae8f4671
----------------------------------------
rand    :193a1b84376c88882c8d3b4ee93ba8d5
PIN length = 3 octets
PIN     :a123b9
address :4459a44610f6
round  1:193a1b84376c88882c8d3b4ee93ba8dc
Key [ 1]:a123b94459a44610f6a123b94459a446
Key [ 2]:5f64d384c8e990c1d25080eb244dde9b
round  2:3badbd58f100831d781ddd3ccedefd3f
Key [ 3]:5abc00eff8991575c00807c48f6dbea5
Key [ 4]:127521158ad6798fb6479d1d2268abe6
round  3:0b53075a49c6bf2df2421c655fdedf68
added ->:128d22de7e3247a5decf572bb61987b4
Key [ 5]:f2a1f620448b8e56665608df2ab3952f
Key [ 6]:7c84c0af02aad91dc39209c4edd220b1
round  4:793f4484fb592e7a78756fd4662f990d
Key [ 7]:f6445b647317e7e493bb92bf6655342f
Key [ 8]:3cae503567c63d3595eb140ce60a84c0
round  5:9e46a8df925916a342f299a8306220a0
Key [ 9]:734ed5a806e072bbecb4254993871679
Key [10]:cda69ccb4b07f65e3c8547c11c0647b8
round  6:6bf9cd82c9e1be13fc58eae0b936c75a
Key [11]:c48e531d3175c2bd26fa25cc8990e394
Key [12]:6d93d349a6c6e9ff5b26149565b13d15
round  7:e96a9871471240f198811d4b8311e9a6
Key [13]:5c4951e85875d663526092cd4cbdb667
Key [14]:f19f7758f5cde86c3791efaf563b3fd0
round  8:e94ca67d3721d5fb08ec069191801a46
Key [15]:bf0c17f3299b37d984ac938b769dd394
Key [16]:7edf4ad772a6b9048588f97be25bde1c
Key [17]:6ee7ba6afefc5b561abbd8d6829e8150
Ka      :ac0daabf17732f632e34ef193658bf5d
----------------------------------------
rand    :1453db4d057654e8eb62d7d62ec3608c
PIN length = 2 octets
PIN     :3eaf
address :411fbbb51d1e
round  1:1453db4d057654e8eb62d7d62ec36084
```

Sample Data

```
Key [ 1]:3eaf411fbbb51d1e3eaf411fbbb51d1e
Key [ 2]:c3a1a997509f00fb4241aba607109c64
round  2:0b78276c1ebc65707d38c9c5fa1372bd
Key [ 3]:3c729833ae1ce7f84861e4dbad6305cc
Key [ 4]:c83a43c3a66595cb8136560ed29be4ff
round  3:23f3f0f6441563d4c202cee0e5cb2335
added ->:3746cbbb418bb73c2964a536cb8e83b1
Key [ 5]:18b26300b86b70acdd1c8f5cbc7c5da8
Key [ 6]:04efc75309b98cd8f1cef5513c18e41e
round  4:c61afa90d3c14bdf588320e857afdc00
Key [ 7]:517c789cecadc455751af73198749fb8
Key [ 8]:fd9711f913b5c844900fa79dd765d0e2
round  5:a8a0e02ceb556af8bfa321789801183a
Key [ 9]:bb5cf30e7d3ceb930651b1d16ee92750
Key [10]:3d97c7862ecab42720e984972f8efd28
round  6:0b58e922438d224db34b68fca9a5ea12
Key [11]:4ce730344f6b09e449dcdb64cd466666
Key [12]:38828c3a56f922186adcd9b713cdcc31
round  7:b90664c4ac29a8b4bb26debec9ffc5f2
Key [13]:d30fd865ea3e9edcff86a33a2c319649
Key [14]:1fdb63e54413acd968195ab6fa424e83
round  8:6934de3067817cefd811abc5736c163b
Key [15]:a16b7c655bbaa262c807cba8ae166971
Key [16]:7903dd68630105266049e23ca607cda7
Key [17]:888446f2d95e6c2d2803e6f4e815ddc9
Ka      :1674f9dc2063cc2b83d3ef8ba692ebef
----------------------------------------
rand    :1313f7115a9db842fcedc4b10088b48d
PIN length = 1 octets
PIN     :6d
address :008aa9be62d5
round  1:1313f7115a9db842fcedc4b10088b48a
Key [ 1]:6d008aa9be62d56d008aa9be62d56d00
Key [ 2]:46ebfeafb6657b0a1984a8dc0893accf
round  2:839b23b83b5701ab095bafd162ec0ac7
Key [ 3]:8e15595edcf058af62498ee3c1dc6098
Key [ 4]:dd409c3444e94b9cc08396ae967542a0
round  3:c0a2010cc44f2139427f093f4f97ae68
added ->:d3b5f81d9eecd97bbe6ccd8e4f1f62e2
Key [ 5]:487deff5d519f6a6481e947b926f633c
Key [ 6]:5b4b6e3477ed5c2c01f6e607d3418963
round  4:1a5517a0efad3575931d8ea3bee8bd07
Key [ 7]:34b980088d2b5fd6b6a2aceeda99c9c4
Key [ 8]:e7d06d06078acc4ecdbc8da800b73078
round  5:d3ce1fdfe716d72c1075ff37a8a2093f
Key [ 9]:7d375bad245c3b757380021af8ecd408
Key [10]:14dac4bc2f4dc4929a6cceec47f4c3a3
round  6:47e90cb55be6e8dd0f583623c2f2257b
Key [11]:66cfda3c63e464b05e2e7e25f8743ad7
Key [12]:77cfccda1ad380b9fdf1df10846b50e7
round  7:f866ae6624f7abd4a4f5bd24b04b6d43
Key [13]:3e11dd84c031a470a8b66ec6214e44cf
```

Sample Data

```
Key [14]:2f03549bdb3c511eea70b65ddbb08253
round  8:02e8e17cf8be4837c9c40706b613dfa8
Key [15]:e2f331229ddfcc6e7bea08b01ab7e70c
Key [16]:b6b0c3738c5365bc77331b98b3fba2ab
Key [17]:f5b3973b636119e577c5c15c87bcfd19
Ka      :38ec0258134ec3f08461ae5c328968a1

========================================
```

9.5 FOUR TESTS OF E3

```
rand    :00000000000000000000000000000000
aco     :48afcdd4bd40fef76693b113
key     :00000000000000000000000000000000
round  1:00000000000000000000000000000000
Key [ 1]:00000000000000000000000000000000
Key [ 2]:4697b1baa3b7100ac537b3c95a28ac64
round  2:78d19f9307d2476a523ec7a8a026042a
Key [ 3]:ecabaac66795580df89af66e66dc053d
Key [ 4]:8ac3d8896ae9364943bfebd4969b68a0
round  3:600265247668dda0e81c07bbb30ed503
Key [ 5]:5d57921fd5715cbb22c1be7bbc996394
Key [ 6]:2a61b8343219fdfb1740e6511d41448f
round  4:d7552ef7cc9dbde568d80c2215bc4277
Key [ 7]:dd0480dee731d67f01a2f739da6f23ca
Key [ 8]:3ad01cd1303e12a1cd0fe0a8af82592c
round  5:fb06bef32b52ab8f2a4f2b6ef7f6d0cd
Key [ 9]:7dadb2efc287ce75061302904f2e7233
Key [10]:c08dcfa981e2c4272f6c7a9f52e11538
round  6:b46b711ebb3cf69e847a75f0ab884bdd
Key [11]:fc2042c708e409555e8c147660ffdfd7
Key [12]:fa0b21001af9a6b9e89e624cd99150d2
round  7:c585f308ff19404294f06b292e978994
Key [13]:18b40784ea5ba4c80ecb48694b4e9c35
Key [14]:454d54e5253c0c4a8b3fcca7db6baef4
round  8:2665fadb13acf952bf74b4ab12264b9f
Key [15]:2df37c6d9db52674f29353b0f011ed83
Key [16]:b60316733b1e8e70bd861b477e2456f1
Key [17]:884697b1baa3b7100ac537b3c95a28ac
round  1:5d3ecb17f26083df0b7f2b9b29aef87c
Key [ 1]:e9e5dfc1b3a79583e9e5dfc1b3a79583
Key [ 2]:7595bf57e0632c59f435c16697d4c864
round  2:de6fe85c5827233fe22514a16f321bd8
Key [ 3]:e31b96afcc75d286ef0ae257cbbc05b7
Key [ 4]:0d2a27b471bc0108c6263aff9d9b3b6b
round  3:7cd335b50d09d139ea6702623af85edb
added ->:211100a2ff6954e6e1e62df913a656a7
Key [ 5]:98d1eb5773cf59d75d3b17b3bc37c191
Key [ 6]:fd2b79282408ddd4ea0aa7511133336f
round  4:991dccb3201b5b1c4ceff65a3711e1e9
Key [ 7]:331227756638a41d57b0f7e071ee2a98
```

Sample Data

```
Key [ 8]:aa0dd8cc68b406533d0f1d64aabacf20
round  5:18768c7964818805fe4c6ecae8a38599
Key [ 9]:669291b0752e63f806fce76f10e119c8
Key [10]:ef8bdd46be8ee0277e9b78adef1ec154
round  6:82f9aa127a72632af43d1a17e7bd3a09
Key [11]:f3902eb06dc409cfd78384624964bf51
Key [12]:7d72702b21f97984a721c99b0498239d
round  7:1543d7870bf2d6d6efab3cbf62dca97d
Key [13]:532e60bceaf902c52a06c2c283ecfa32
Key [14]:181715e5192efb2a64129668cf5d9dd4
round  8:eee3e8744a5f8896de95831ed837ffd5
Key [15]:83017c1434342d4290e961578790f451
Key [16]:2603532f365604646ff65803795ccce5
Key [17]:882f7c907b565ea58dae1c928a0dcf41
kc      :cc802aecc7312285912e90af6a1e1154
----------------------------------------
rand    :950e604e655ea3800fe3eb4a28918087
aco     :68f4f472b5586ac5850f5f74
key     :34e86915d20c485090a6977931f96df5
round  1:950e604e655ea3800fe3eb4a28918087
Key [ 1]:34e86915d20c485090a6977931f96df5
Key [ 2]:8de2595003f9928efaf37e5229935bdb
round  2:d46f5a04c967f55840f83d1cdb5f9afc
Key [ 3]:46f05ec979a97cb6ddf842ecc159c04a
Key [ 4]:b468f0190a0a83783521deae8178d071
round  3:e16ededе9cb6297f32e1203e442ac73a
Key [ 5]:8a171624dedbd552356094daaadcf12a
Key [ 6]:3085e07c85e4b99313f6e0c837b5f819
round  4:805144e55e1ece96683d23366fc7d24b
Key [ 7]:fe45c27845169a66b679b2097d147715
Key [ 8]:44e2f0c35f64514e8bec66c5dc24b3ad
round  5:edbaf77af070bd22e9304398471042f1
Key [ 9]:0d534968f3803b6af447eaf964007e7b
Key [10]:f5499a32504d739ed0b3c547e84157ba
round  6:0dab1a4c846aef0b65b1498812a73b50
Key [11]:e17e8e456361c46298e6592a6311f3fb
Key [12]:ec6d14da05d60e8abac807646931711f
round  7:1e7793cac7f55a8ab48bd33bc9c649e0
Key [13]:2b53dde3d89e325e5ff808ed505706ae
Key [14]:41034e5c3fb0c0d4f445f0cf23be79b0
round  8:3723768baa78b6a23ade095d995404da
Key [15]:e2ca373d405a7abf22b494f28a6fd247
Key [16]:74e09c9068c0e8f1c6902d1b70537c30
Key [17]:767a7f1acf75c3585a55dd4a428b2119
round  1:39809afb773efd1b7510cd4cb7c49f34
Key [ 1]:1d0d48d485abddd3798b483a82a0f878
Key [ 2]:aed957e600a5aed5217984dd5fef6fd8
round  2:6436ddbabe92655c87a7d0c12ae5e5f6
Key [ 3]:fee00bb0de89b6ef0a289696a4faa884
Key [ 4]:33ce2f4411db4dd9b7c42cc586b8a2ba
round  3:cec690f7e0aa5f063062301e049a5cc5
added ->:f7462a0c97e85c1d4572fd52b35efbf1
```

Sample Data

```
Key [ 5]:b5116f5c6c29e05e4acb4d02a46a3318
Key [ 6]:ff4fa1f0f73d1a3c67bc2298abc768f9
round  4:dcdfe942e9f0163fc24a4718844b417d
Key [ 7]:5453650c0819e001e48331ad0e9076e0
Key [ 8]:b4ff8dda778e26c0dce08349b81c09a1
round  5:265a16b2f766afae396e7a98c189fda9
Key [ 9]:f638fa294427c6ed94300fd823b31d10
Key [10]:1ccfa0bd86a9879b17d4bc457e3e03d6
round  6:628576b5291d53d1eb8611c8624e863e
Key [11]:0eaee2ef4602ac9ca19e49d74a76d335
Key [12]:6e1062f10a16e0d378476da3943842e9
round  7:d7b9c2e9b2d5ea5c27019324cae882b3
Key [13]:40be960bd22c744c5b23024688e554b9
Key [14]:95c9902cb3c230b44d14ba909730d211
round  8:97fb6065498385e47eb3df6e2ca439dd
Key [15]:10d4b6e1d1d6798aa00aa2951e32d58d
Key [16]:c5d4b91444b83ee578004ab8876ba605
Key [17]:1663a4f98e2862eddd3ec2fb03dcc8a4
kc      :c1beafea6e747e304cf0bd7734b0a9e2
----------------------------------------
rand    :6a8ebcf5e6e471505be68d5eb8a3200c
aco     :658d791a9554b77c0b2f7b9f
key     :35cf77b333c294671d426fa79993a133
round  1:6a8ebcf5e6e471505be68d5eb8a3200c
Key [ 1]:35cf77b333c294671d426fa79993a133
Key [ 2]:c4524e53b95b4bf2d7b2f095f63545fd
round  2:ade94ec585db0d27e17474b58192c87a
Key [ 3]:c99776768c6e9f9dd3835c52cea8d18a
Key [ 4]:f1295db23823ba2792f21217fc01d23f
round  3:da8dc1a10241ef9e6e069267cd2c6825
Key [ 5]:9083db95a6955235bbfad8aeefec5f0b
Key [ 6]:8bab6bc253d0d0c7e0107feab728ff68
round  4:e6665ca0772ceecbc21222ff7be074f8
Key [ 7]:2fa1f4e7a4cf3ccd876ec30d194cf196
Key [ 8]:267364be247184d5337586a19df8bf84
round  5:a857a9326c9ae908f53fee511c5f4242
Key [ 9]:9aef21965b1a6fa83948d107026134c7
Key [10]:d2080c751def5dc0d8ea353cebf7b973
round  6:6678748a1b5f21ac05cf1b117a7c342f
Key [11]:d709a8ab70b0d5a2516900421024b81e
Key [12]:493e4843805f1058d605c8d1025f8a56
round  7:766c66fe9c460bb2ae39ec01e435f725
Key [13]:b1ed21b71daea03f49fe74b2c11fc02b
Key [14]:0e1ded7ebf23c72324a0165a698c65c7
round  8:396e0ff7b2b9b7a3b35c9810882c7596
Key [15]:b3bf4841dc92f440fde5f024f9ce8be9
Key [16]:1c69bc6c2994f4c84f72be8f6b188963
Key [17]:bb7b66286dd679a471e2792270f3bb4d
round  1:45654f2f26549675287200f07cb10ec9
Key [ 1]:1e2a5672e66529e4f427b0682a3a34b6
Key [ 2]:974944f1ce0037b1febcf61a2bc961a2
round  2:990cd869c534e76ed4f4af7b3bfbc6c8
```

Sample Data

```
Key [ 3]:8147631fb1ce95d624b480fc7389f6c4
Key [ 4]:6e90a2db33d284aa13135f3c032aa4f4
round  3:ceb662f875aa6b94e8192b5989abf975
added ->:8b1bb1d753fe01e1c08b2ba9f55c07bc
Key [ 5]:cbad246d24e36741c46401e6387a05f9
Key [ 6]:dcf52aaec5713110345a41342c566fc8
round  4:d4e000be5de78c0f56ff218f3c1df61b
Key [ 7]:8197537aa9d27e67d17c16b182c8ec65
Key [ 8]:d66e00e73d835927a307a3ed79d035d8
round  5:9a4603bdef954cfaade2052604bed4e4
Key [ 9]:71d46257ecc1022bcd312ce6c114d75c
Key [10]:f91212fa528379651fbd2c32890c5e5f
round  6:09a0fd197ab81eb933eece2fe0132dbb
Key [11]:283acc551591fadce821b02fb9491814
Key [12]:ca5f95688788e20d94822f162b5a3920
round  7:494f455a2e7a5db861ece816d4e363e4
Key [13]:ba574aef663c462d35399efb999d0e40
Key [14]:6267afc834513783fef1601955fe0628
round  8:37a819f91c8380fb7880e640e99ca947
Key [15]:fdcd9be5450eef0f8737e6838cd38e2b
Key [16]:8cfbd9b8056c6a1ce222b92b94319b38
Key [17]:4f64c1072c891c39eeb95e63318462e0
kc      :a3032b4df1cceba8adc1a04427224299
----------------------------------------
rand    :5ecd6d75db322c75b6afbd799cb18668
aco     :63f701c7013238bbf88714ee
key     :b9f90c53206792b1826838b435b87d4d
round  1:5ecd6d75db322c75b6afbd799cb18668
Key [ 1]:b9f90c53206792b1826838b435b87d4d
Key [ 2]:15f74bbbde4b9d1e08f858721f131669
round  2:72abb85fc80c15ec2b00d72873ef9ad4
Key [ 3]:ef7fb29f0b01f82706c7439cc52f2dab
Key [ 4]:3003a6aecdee06b9ac295cce30dcdb93
round  3:2f10bab93a0f73742183c68f712dfa24
Key [ 5]:5fcdbb3afdf7df06754c954fc6340254
Key [ 6]:ddaa90756635579573fe8ca1f93d4a38
round  4:183b145312fd99d5ad08e7ca4a52f04e
Key [ 7]:27ca8a7fc703aa61f6d7791fc19f704a
Key [ 8]:702029d8c6e42950762317e730ec5d18
round  5:cbad52d3a026b2e38b9ae6fefffecc32
Key [ 9]:ff15eaa3f73f4bc2a6ccfb9ca24ed9c5
Key [10]:034e745246cd2e2cfc3bda39531ca9c5
round  6:ce5f159d0a1acaacd9fb4643272033a7
Key [11]:0a4d8ff5673731c3dc8fe87e39a34b77
Key [12]:637592fab43a19ac0044a21afef455a2
round  7:8a49424a10c0bea5aba52dbbffcbcce8
Key [13]:6b3fde58f4f6438843cdbe92667622b8
Key [14]:a10bfa35013812f39bf2157f1c9fca4e
round  8:f5e12da0e93e26a5850251697ec0b917
Key [15]:2228fe5384e573f48fdd19ba91f1bf57
Key [16]:5f174db2bc88925c0fbc6b5485bafc08
Key [17]:28ff90bd0dc31ea2bb479feb7d8fe029
```

Sample Data

```
round  1:0c75eed2b54c1cfb9ff522daef94ed4d
Key [ 1]:a21ceb92d3c027326b4de775865fe8d0
Key [ 2]:26f64558a9f0a1652f765efd546f3208
round  2:48d537ac209a6aa07b70000016c602e8
Key [ 3]:e64f9ef630213260f1f79745a0102ae5
Key [ 4]:af6a59d7cebfd0182dcca9a537c4add8
round  3:8b6d517ac893743a401b3fb7911b64e1
added ->:87e23fa87ddf90c1df10616d7eaf51ac
Key [ 5]:9a6304428b45da128ab64c8805c32452
Key [ 6]:8af4d1e9d80cb73ec6b44e9b6e4f39d8
round  4:9f0512260a2f7a5067efc35bf1706831
Key [ 7]:79cc2d138606f0fca4e549c34a1e6d19
Key [ 8]:803dc5cdde0efdbee7a1342b2cd4d344
round  5:0cfd7856edfafac51f29e86365de6f57
Key [ 9]:e8fa996448e6b6459ab51e7be101325a
Key [10]:2acc7add7b294acb444cd933f0e74ec9
round  6:2f1fa34bf352dc77c0983a01e8b7d622
Key [11]:f57de39e42182efd6586b86a90c86bb1
Key [12]:e418dfd1bb22ebf1bfc309cd27f5266c
round  7:ee4f7a53849bf73a747065d35f3752b1
Key [13]:80a9959133856586370854db6e0470b3
Key [14]:f4c1bc2f764a0193749f5fc09011a1ae
round  8:8fec6f7249760ebf69e370e9a4b80a92
Key [15]:d036cef70d6470c3f52f1b5d25b0c29d
Key [16]:d0956af6b8700888a1cc88f07ad226dc
Key [17]:1ce8b39c4c7677373c30849a3ee08794
kc      :ea520cfc546b00eb7c3a6cea3ecb39ed

========================================
```

Core System Package [Controller volume]
Part H



SECURITY SPECIFICATION

This document describes the specification of the security system which may be used at the link layer. The Encryption, Authentication and Key Generation schemes are specified. The requirements for the supporting process of random number generation are also specified.

CONTENTS

1 Security Overview 773

2 Random Number Generation 775

3 Key Management 777

3.1 Key Types 777

3.2 Key Generation and Initialization 779

3.2.1 Generation of initialization key, ... 780

3.2.2 Authentication 780

3.2.3 Generation of a unit key 780

3.2.4 Generation of a combination key 781

3.2.5 Generating the encryption key 782

3.2.6 Point-to-multipoint configuration 783

3.2.7 Modifying the link keys 784

3.2.8 Generating a master key 784

4 Encryption 787

4.1 Encryption Key Size Negotiation 788

4.2 Encryption of Broadcast Messages 788

4.3 Encryption Concept 789

4.4 Encryption Algorithm 790

4.4.1 The operation of the cipher 792

4.5 LFSR Initialization 793

4.6 Key Stream Sequence 796

5 Authentication 797

5.1 Repeated Attempts 799

6 The Authentication And Key-Generating Functions 801

6.1 The Authentication Function E1 801

6.2 The Functions Ar and A'r 803

6.2.1 The round computations 803

6.2.2 The substitution boxes "e" and "l" 803

6.2.3 Key scheduling 804

6.3 E2-Key Generation Function for Authentication 805

6.4 E3-Key Generation Function for Encryption 807

7 List of Figures 809

8 List of Tables 811

1 SECURITY OVERVIEW

Bluetooth wireless technology provides peer-to-peer communications over short distances. In order to provide usage protection and information confidentiality, the system provides security measures both at the application layer and the link layer. These measures are designed to be appropriate for a peer environment. This means that in each device, the authentication and encryption routines are implemented in the same way. Four different entities are used for maintaining security at the link layer: a Bluetooth device address, two secret keys, and a pseudo-random number that shall be regenerated for each new transaction. The four entities and their sizes are summarized in Table 1.1.

Entity	Size
BD_ADDR	48 bits
Private user key, authentication	128 bits
Private user key, encryption configurable length (byte-wise)	8-128 bits
RAND	128 bits

Table 1.1: Entities used in authentication and encryption procedures.

The Bluetooth device address (BD_ADDR) is the 48-bit address. The BD_ADDR can be obtained via user interactions, or, automatically, via an inquiry routine by a device.

The secret keys are derived during initialization and are never disclosed. The encryption key is derived from the authentication key during the authentication process. For the authentication algorithm, the size of the key used is always 128 bits. For the encryption algorithm, the key size may vary between 1 and 16 octets (8 - 128 bits). The size of the encryption key is configurable for two reasons. The first has to do with the many different requirements imposed on cryptographic algorithms in different countries – both with respect to export regulations and official attitudes towards privacy in general. The second reason is to facilitate a future upgrade path for the security without the need of a costly redesign of the algorithms and encryption hardware; increasing the effective key size is the simplest way to combat increased computing power at the opponent side.

The encryption key is entirely different from the authentication key (even though the latter is used when creating the former, as is described in Section 6.4 on page 807). Each time encryption is activated, a new encryption key shall be generated. Thus, the lifetime of the encryption key does not necessarily correspond to the lifetime of the authentication key.

It is anticipated that the authentication key will be more static in its nature than the encryption key – once established, the particular application running on the device decides when, or if, to change it. To underline the fundamental impor-

tance of the authentication key to a specific link, it is often be referred to as the link key.

The RAND is a pseudo-random number which can be derived from a random or pseudo-random process in the device. This is not a static parameter, it will change frequently.

In the remainder of this chapter, the terms user and application will be used interchangeably to designate the entity that is at either side.

2 RANDOM NUMBER GENERATION

Each device has a pseudo-random number generator. Pseudo-random numbers are used for many purposes within the security functions – for instance, for the challenge-response scheme, for generating authentication and encryption keys, etc. Ideally, a true random generator based on some physical process with inherent randomness should be used as a seed. Examples of such processes are thermal noise from a semiconductor or resistor and the frequency instability of a free running oscillator. For practical reasons, a software based solution with a pseudo-random generator is probably preferable. In general, it is quite difficult to classify the randomness of a pseudo-random sequence. Within this specification, the requirements placed on the random numbers used are non-repeating and randomly generated.

The expression 'non-repeating' means that it shall be highly unlikely that the value will repeat itself within the lifetime of the authentication key. For example, a non-repeating value could be the output of a counter that is unlikely to repeat during the lifetime of the authentication key, or a date/time stamp.

The expression 'randomly generated' means that it shall not be possible to predict its value with a chance that is significantly larger than 0 (e.g., greater than $1/2^L$ for a key length of L bits).

The LM may use such a generator for various purposes; i.e. whenever a random number is needed (such as the RANDs, the unit keys, K_{init}, K_{master}, and random back-off or waiting intervals).

3 KEY MANAGEMENT

It is important that the encryption key size within a specific device cannot be set by the user – this should be a factory preset entity. In order to prevent the user from over-riding the permitted key size, the Bluetooth baseband processing shall not accept an encryption key given from higher software layers. Whenever a new encryption key is required, it shall be created as defined in Section 6.4 on page 807.

Changing a link key shall also be done through the defined baseband procedures. Depending on what kind of link key it is, different approaches are required. The details are found in Section 3.2.7 on page 784.

3.1 KEY TYPES

The link key is a 128-bit random number which is shared between two or more parties and is the base for all security transactions between these parties. The link key itself is used in the authentication routine. Moreover, the link key is used as one of the parameters when the encryption key is derived.

In the following, a session is defined as the time interval for which the device is a member of a particular piconet. Thus, the session terminates when the device disconnects from the piconet.

The link keys are either semi-permanent or temporary. A semi-permanent link key may be stored in non-volatile memory and may be used after the current session is terminated. Consequently, once a semi-permanent link key is defined, it may be used in the authentication of several subsequent connections between the devices sharing it. The designation semi-permanent is justified by the possibility of changing it. How to do this is described in Section 3.2.7 on page 784.

The lifetime of a temporary link key is limited by the lifetime of the current session – it shall not be reused in a later session. Typically, in a point-to-multipoint configuration where the same information is to be distributed securely to several recipients, a common encryption key is useful. To achieve this, a special link key (denoted master key) may temporarily replace the current link keys. The details of this procedure are found in Section 3.2.6 on page 783.

In the following, the current link key is the link key in use at the current moment. It can be semi-permanent or temporary. Thus, the current link key is used for all authentications and all generation of encryption keys in the on-going connection (session).

In order to accommodate different types of applications, four types of link keys have been defined:

- the combination key K_{AB}
- the unit key K_A
- the temporary key K_{master}
- the initialization key K_{init}

Note: the use of unit keys is deprecated since it is implicitly insecure.

In addition to these keys there is an encryption key, denoted K_c. This key is derived from the current link key. Whenever encryption is activated by an LM command, the encryption key shall be changed automatically. The purpose of separating the authentication key and encryption key is to facilitate the use of a shorter encryption key without weakening the strength of the authentication procedure. There are no governmental restrictions on the strength of authentication algorithms. However, in some countries, such restrictions exist on the strength of encryption algorithms.

The combination key K_{AB} and the unit key K_A are functionally indistinguishable; the difference is in the way they are generated. The unit key K_A is generated in, and therefore dependent on, a single device A. The unit key shall be generated once at installation of the device; thereafter, it is very rarely changed. The combination key is derived from information in both devices A and B, and is therefore always dependent on two devices. The combination key is derived for each new combination of two devices.

It depends on the application or the device whether a unit key or a combination key is used. Devices which have little memory to store keys, or are installed in equipment that will be accessible to a large group of users, should use their own unit key. In that case, they only have to store a single key. Applications that require a higher security level should use the combination keys. These applications will require more memory since a combination key for each link to a different device has to be stored.

The master key, K_{master}, shall only be used during the current session. It shall only replace the original link key temporarily. For example, this may be utilized when a master wants to reach more than two devices simultaneously using the same encryption key, see Section 3.2.6 on page 783.

The initialization key, K_{init}, shall be used as the link key during the initialization process when no combination or unit keys have been defined and exchanged yet or when a link key has been lost. The initialization key protects the transfer of initialization parameters. The key is derived from a random number, an L-octet PIN code, and a BD_ADDR. This key shall only be used during initialization.

The PIN may be a fixed number provided with the device (for example when there is no user interface as in a PSTN plug). Alternatively, the PIN can be selected by the user, and then entered in both devices that are to be matched. The latter procedure should be used when both devices have a user interface, for example a phone and a laptop. Entering a PIN in both devices is more secure than using a fixed PIN in one of the devices, and should be used whenever possible. Even if a fixed PIN is used, it shall be possible to change the PIN; this is in order to prevent re-initialization by users who once got hold of the PIN. If no PIN is available, a default value of zero may be used. The length of this default PIN is one byte, PIN(default) = 0x00. This default PIN may be provided by the host.

For many applications the PIN code will be a relatively short string of numbers. Typically, it may consist of only four decimal digits. Even though this gives sufficient security in many cases, there exist countless other, more sensitive, situations where this is not reliable enough. Therefore, the PIN code may be chosen to be any length from 1 to 16 octets. For the longer lengths, the devices exchanging PIN codes may not use mechanical (i.e. human) interaction, but rather may use software at the application layer. For example, this can be a Diffie-Hellman key agreement, where the exchanged key is passed on to the K_{init} generation process in both devices, just as in the case of a shorter PIN code.

3.2 KEY GENERATION AND INITIALIZATION

The link keys must be generated and distributed among the devices in order to be used in the authentication procedure. Since the link keys shall be secret, they shall not be obtainable through an inquiry routine in the same way as the Bluetooth device addresses. The exchange of the keys takes place during an initialization phase which shall be carried out separately for each two devices that are using authentication and encryption. The initialization procedures consist of the following five parts:

- generation of an initialization key
- generation of link key
- link key exchange
- authentication
- generation of encryption key in each device (optional)

After the initialization procedure, the devices can proceed to communicate, or the link can be disconnected. If encryption is implemented, the E_0 algorithm shall be used with the proper encryption key derived from the current link key. For any new connection established between devices A and B, they should use the common link key for authentication, instead of once more deriving K_{init} from the PIN. A new encryption key derived from that particular link key shall be created next time encryption is activated.

If no link key is available, the LM shall automatically start an initialization procedure.

3.2.1 Generation of initialization key, K_{init}

A link key is used temporarily during initialization, the initialization key K_{init}. This key shall be derived by the E_{22} algorithm from a BD_ADDR, a PIN code, the length of the PIN (in octets), and a random number IN_RAND. The principle is depicted in Figure 6.4 on page 807. The 128-bit output from E_{22} shall be used for key exchange during the generation of a link key. When the devices have performed the link key exchange, the initialization key shall be discarded.

When the initialization key is generated, the PIN is augmented with the BD_ADDR. If one device has a fixed PIN the BD_ADDR of the other device shall be used. If both devices have a variable PIN the BD_ADDR of the device that received IN_RAND shall be used. If both devices have a fixed PIN they cannot be paired. Since the maximum length of the PIN used in the algorithm cannot exceed 16 octets, it is possible that not all octets of BD_ADDR will be used. This procedure ensures that K_{init} depends on the identity of the device with a variable PIN. A fraudulent device may try to test a large number of PINs by claiming another BD_ADDR each time. It is the application's responsibility to take countermeasures against this threat. If the device address is kept fixed, the waiting interval before the next try may be increased exponentially (see Section 5.1 on page 799).

The details of the E_{22} algorithm can be found in Section 6.3 on page 805.

3.2.2 Authentication

The authentication procedure shall be carried out as described in Section 5 on page 797. During each authentication, a new AU_RAND_A shall be issued.

Mutual authentication is achieved by first performing the authentication procedure in one direction and then immediately performing the authentication procedure in the opposite direction.

As a side effect of a successful authentication procedure an auxiliary parameter, the Authenticated Ciphering Offset (ACO), will be computed. The ACO shall be used for ciphering key generation as described in Section 3.2.5 on page 782.

The claimant/verifier status is determined by the LM.

3.2.3 Generation of a unit key

A unit key K_A shall be generated when the device is in operation for the first time; i.e. not during each initialization. The unit key shall be generated by the E_{21} algorithm as described in Section 6.3 on page 805. Once created, the unit key should be stored in non-volatile memory and very rarely changed. If after initialization

the unit key is changed, any previously initialized devices will possess a wrong link key. At initialization, the application must determine which of the two parties will provide the unit key as the link key. Typically, this will be the device with restricted memory capabilities, since this device only has to remember its own unit key. The unit key shall be transferred to the other party and then stored as the link key for that particular party. So, for example in Figure 3.1 on page 781, the unit key of device A, K_A, is being used as the link key for the connection A-B; device A sends the unit key K_A to device B; device B will store K_A as the link key K_{BA}. For another initialization, for example with device C, device A will reuse its unit key K_A, whereas device C stores it as K_{CA}.

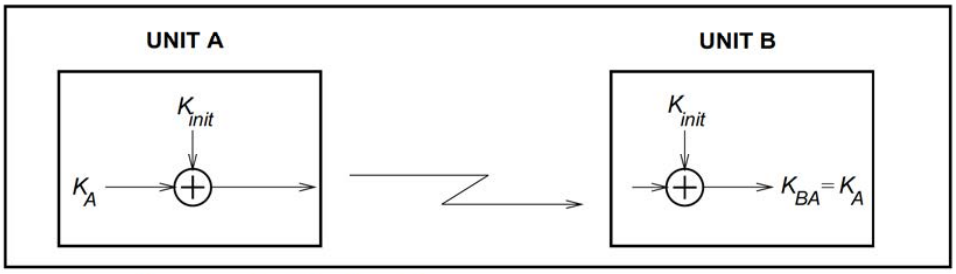


Figure 3.1: Generation of unit key. When the unit key has been exchanged, the initialization key is discarded in both devices.

3.2.4 Generation of a combination key

To use a combination key, it is first generated during the initialization procedure. The combination key is the combination of two numbers generated in device A and B, respectively. First, each device shall generate a random number, LK_RAND_A and LK_RAND_B. Then, utilizing E_{21} with the random number and their own BD_ADDRs, the two random numbers

$$LK\_K_A = E_{21}(LK\_RAND_A, BD\_ADDR_A), \quad \text{(EQ 1)}$$

and

$$LK\_K_B = E_{21}(LK\_RAND_B, BD\_ADDR_B), \quad \text{(EQ 2)}$$

shall be created in device A and device B, respectively. These numbers constitute the devices' contribution to the combination key that is to be created. Then, the two random numbers LK_RAND_A and LK_RAND_B shall be exchanged securely by XORing with the current link key, K. Thus, device A shall send $K \oplus LK\_RAND_A$ to device B, while device B shall send $K \oplus LK\_RAND_B$ to device A. If this is done during the initialization phase the link key $K = K_{init}$.

When the random numbers LK_RAND_A and LK_RAND_B have been mutually exchanged, each device shall recalculate the other device's contribution to the combination key. This is possible since each device knows the Bluetooth device address of the other device. Thus, device A shall calculate (EQ 2) on page 781 and device B shall calculate (EQ 1) on page 781. After this, both devices shall combine the two numbers to generate the 128-bit link key. The combining operation is a simple bitwise modulo-2 addition (i.e. XOR). The result shall be stored in device A as the link key K_{AB} and in device B as the link key K_{BA}. When both devices have derived the new combination key, a mutual authentication procedure shall be initiated to confirm the success of the transaction. The old link key shall be discarded after a successful exchange of a new combination key. The message flow between master and slave and the principle for creating the combination key is depicted in Figure 3.2 on page 782.

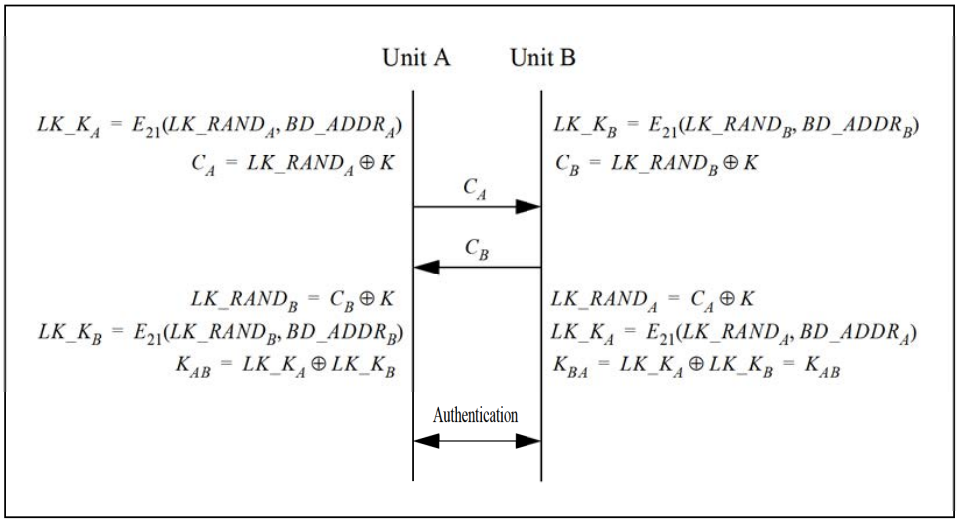


Figure 3.2: Generating a combination key. The old link key (K) is discarded after the exchange of a new combination key has succeeded

3.2.5 Generating the encryption key

The encryption key, K_C, is derived by algorithm E_3 from the current link key, a 96-bit Ciphering OFfset number (COF), and a 128-bit random number. The COF is determined in one of two ways. If the current link key is a master key, then COF shall be derived from the master BD_ADDR. Otherwise the value of COF shall be set to the value of ACO as computed during the authentication procedure. Therefore:[1]

1. $x \cup y$ denotes the concatenation of x and y.

$$COF = \begin{cases} BD\_ADDR \cup BD\_ADDR, & \text{if link key is a master key} \\ ACO, & \text{otherwise.} \end{cases} \quad \text{(EQ 3)}$$

There is an explicit call of E_3 when the LM activates encryption. Consequently, the encryption key is automatically changed each time the device enters encryption mode. The details of the key generating function E_3 can be found in Section 6.4 on page 807.

3.2.6 Point-to-multipoint configuration

It is possible for the master to use separate encryption keys for each slave in a point-to-multipoint configuration with ciphering activated. Then, if the application requires more than one slave to listen to the same payload, each slave must be addressed individually. This can cause unwanted capacity loss for the piconet. Moreover, a slave might not be capable of switching between two or more encryption keys in real time (e.g., after looking at the LT_ADDR in the header). Thus, the master cannot use different encryption keys for broadcast messages and individually addressed traffic. Therefore, the master may tell several slave devices to use a common link key (and, hence, indirectly also to use a common encryption key) and may then broadcast the information encrypted. For many applications, this key is only of temporary interest. In the following discussion, this key is denoted by K_{master}.

The transfer of necessary parameters shall be protected by the routine described in Section 3.2.8 on page 784. After the confirmation of successful reception in each slave, the master shall issue a command to the slaves to replace their respective current link key by the new (temporary) master key. Before encryption can be activated, the master shall also generate and distribute a common EN_RAND to all participating slaves. Using this random number and the newly derived master key, each slave shall generate a new encryption key.

Note that the master must negotiate the encryption key length to use individually with each slave that will use the master key. If the master has already negotiated with some of these slaves, it has knowledge of the sizes that can be accepted. There may be situations where the permitted key lengths of some devices are incompatible. In that case, the master must exclude the limiting device from the group.

When all slaves have received the necessary data, the master can communicate information on the piconet securely using the encryption key derived from the new temporary link key. Each slave in possession of the master key can eavesdrop on all encrypted traffic, not only the traffic intended for itself. The master may tell all participants to fall back to their old link keys simultaneously.

3.2.7 Modifying the link keys

A link key based on a unit key can be changed. The unit key is created once during first use. Typically, the link key should be changed rather than the unit key, as several devices may share the same unit key as link key (e.g. a printer whose unit key is distributed to all users using the printer's unit key as link key). Changing the unit key will require re-initialization of all devices connecting. Changing the unit key can be justified in some circumstances, e.g. to deny access to all previously allowed devices.

If the key change concerns combination keys, then the procedure is straightforward. The change procedure is identical to the procedure described in Figure 3.2 on page 782, using the current value of the combination key as link key. This procedure can be carried out at any time after the authentication and encryption start. Since the combination key corresponds to a single link, it can be modified each time this link is established. This will improve the security of the system since then old keys lose their validity after each session.

Starting up an entirely new initialization procedure is also possible. In that case, user interaction is necessary since a PIN will be required in the authentication and encryption procedures.

3.2.8 Generating a master key

The key-change routines described so far are semi-permanent. To create the master link key, which can replace the current link key during a session (see Section 3.2.6 on page 783), other means are needed. First, the master shall create a new link key from two 128-bit random numbers, RAND1 and RAND2. This shall be done by

$$K_{master} = E_{22}(\text{RAND1}, \text{RAND2}, 16). \quad \text{(EQ 4)}$$

This key is a 128-bit random number. The reason for using the output of E_{22} and not directly choosing a random number as the key, is to avoid possible problems with degraded randomness due to a poor implementation of the random number generator within the device.

Then, a third random number, RAND, shall be transmitted to the slave. Using E_{22} with the current link key and RAND as inputs, both the master and the slave shall compute a 128-bit overlay. The master shall send the bitwise XOR of the overlay and the new link key to the slave. The slave, who knows the overlay, shall recalculate K_{master}. To confirm the success of this transaction, the devices shall perform a mutual authentication procedure using the new link key. This procedure shall then be repeated for each slave that receives the new link key. The ACO values from the authentications shall not replace the current ACO, as this ACO is needed to (re)compute a ciphering key when the master falls back to the previous (non-temporary) link key.

The master activates encryption by an LM command. Before activating encryption, the master shall ensure that all slaves receive the same random number, EN_RAND, since the encryption key is derived through the means of E_3 individually in all participating devices. Each slave shall compute a new encryption key as follows:

$$K_C = E_3(K_{master}, \text{EN\_RAND}, \text{COF}) \quad \text{(EQ 5)}$$

where the value of COF shall be derived from the master's BD_ADDR as specified by equation (EQ 3) on page 783. The details of the encryption key generating function are described in Section 6.4 on page 807. The message flow between the master and the slave when generating the master key is depicted in Figure 3.3. Note that in this case the ACO produced during the authentication is not used when computing the ciphering key.

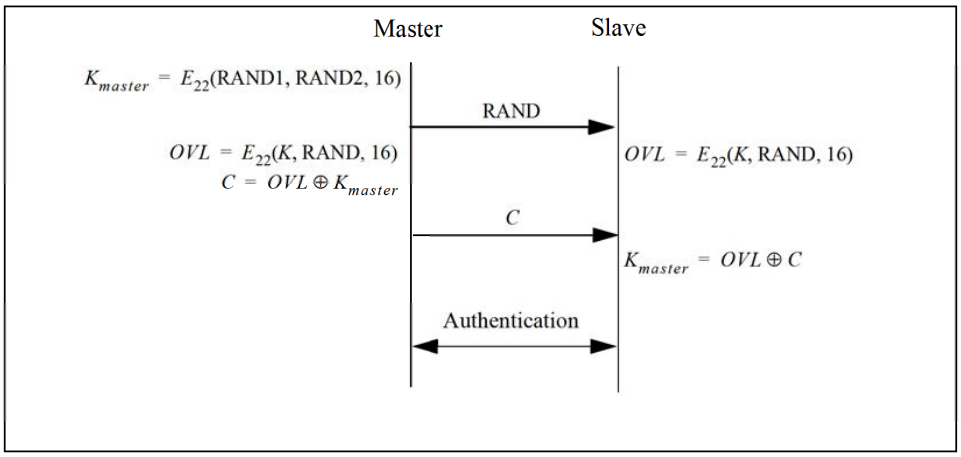


Figure 3.3: Master link key distribution and computation of the corresponding encryption key.

4 ENCRYPTION

User information can be protected by encryption of the packet payload; the access code and the packet header shall never be encrypted. The encryption of the payload shall be carried out with a stream cipher called E_0 that shall be re-synchronized for every payload. The overall principle is shown in Figure 4.1 on page 787.

The stream cipher system E_0 shall consist of three parts:

- the first part performs initialization (generation of the payload key). The payload key generator shall combine the input bits in an appropriate order and shall shift them into the four LFSRs used in the key stream generator.
- the second part generates the key stream bits and shall use a method derived from the summation stream cipher generator attributable to Massey and Rueppel. The second part is the main part of the cipher system, as it will also be used for initialization.
- the third part performs encryption and decryption.

The Massey and Rueppel method has been thoroughly investigated, and there exist good estimates of its strength with respect to presently known methods for cryptanalysis. Although the summation generator has weaknesses that can be used in correlation attacks, the high re-synchronization frequency will disrupt such attacks.

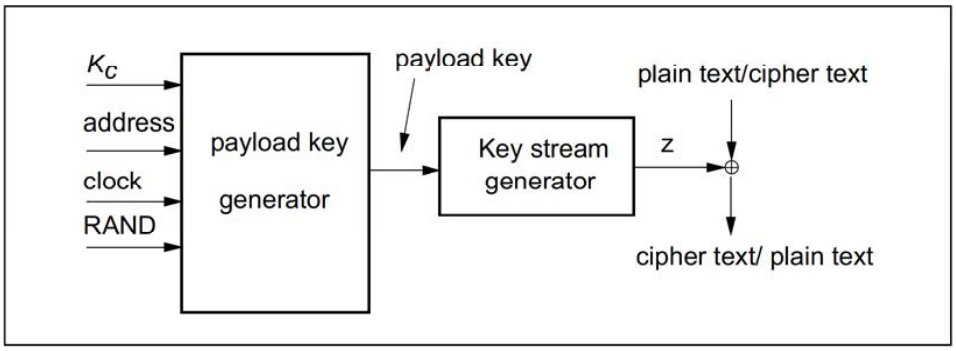


Figure 4.1: Stream ciphering for Bluetooth with E_0.

4.1 ENCRYPTION KEY SIZE NEGOTIATION

Each device implementing the baseband specification shall have a parameter defining the maximal allowed key length, L_{max}, $1 \le L_{max} \le 16$ (number of octets in the key). For each application using encryption, a number L_{min} shall be defined indicating the smallest acceptable key size for that particular application. Before generating the encryption key, the devices involved shall negotiate to decide the key size to use.

The master shall send a suggested value, $L_{sug}^{(M)}$, to the slave. Initially, the suggested value shall be set to $L_{max}^{(M)}$. If $L_{min}^{(S)} \le L_{sug}^{(M)}$, and, the slave supports the suggested length, the slave shall acknowledge and this value shall be the length of the encryption key for this link. However, if both conditions are not fulfilled, the slave shall send a new proposal, $L_{sug}^{(S)} < L_{sug}^{(M)}$, to the master. This value shall be the largest among all supported lengths less than the previous master suggestion. Then, the master shall perform the corresponding test on the slave suggestion. This procedure shall be repeated until a key length agreement is reached, or, one device aborts the negotiation. An abort may be caused by lack of support for L_{sug} and all smaller key lengths, or if $L_{sug} < L_{min}$ in one of the devices. In case of an abort link encryption can not be employed.

The possibility of a failure in setting up a secure link is an unavoidable consequence of letting the application decide whether to accept or reject a suggested key size. However, this is a necessary precaution. Otherwise a fraudulent device could enforce a weak protection on a link by claiming a small maximum key size.

4.2 ENCRYPTION OF BROADCAST MESSAGES

There may be three settings for the baseband regarding encryption:

1. No encryption.
   This is the default setting. No messages are encrypted
2. Point-to-point only encryption.
   Broadcast messages are not encrypted. This may be enabled either during the connection establishment procedure or after the connection has been established.
3. Point-to-point and broadcast encryption.
   All messages are encrypted. This may be enabled after the connection has been established only. This setting should not be enabled unless all affected links share the same master link key as well as the same EN_RAND value, both used in generating the encryption key.

4.3 ENCRYPTION CONCEPT

Broadcast traffic	Individually addressed traffic
No encryption	No encryption
No encryption	Encryption, K_{master}
Encryption, K_{master}	Encryption, K_{master}

Table 4.1: Possible encryption modes for a slave in possession of a master key.

For the encryption routine, a stream cipher algorithm is used in which ciphering bits are bit-wise modulo-2 added to the data stream to be sent over the air interface. The payload is ciphered after the CRC bits are appended, but, prior to the FEC encoding.

Each packet payload shall be ciphered separately. The cipher algorithm E_0 uses the master Bluetooth device address (BD_ADDR), 26 bits of the master real-time clock (CLK $_{26\text{-}1}$) and the encryption key K_C as input, see Figure 4.2 on page 790 (where it is assumed that device A is the master).

The encryption key K_C is derived from the current link key, COF, and a random number, EN_RAND_A (see Section 6.4 on page 807). The random number shall be issued by the master before entering encryption mode. Note that EN_RAND_A is publicly known since it is transmitted as plain text over the air.

Within the E_0 algorithm, the encryption key K_C is modified into another key denoted K'_C. The maximum effective size of this key shall be factory preset and may be set to any multiple of eight between one and sixteen (8-128 bits). The procedure for deriving the key is described in Section 4.5 on page 793.

The real-time clock is incremented for each slot. The E_0 algorithm shall be re-initialized at the start of each new packet (i.e. for Master-to-Slave as well as for Slave-to-Master transmission). By using $CLK_{26\text{-}1}$ at least one bit is changed between two transmissions. Thus, a new keystream is generated after each re-initialization. For packets covering more than a single slot, the Bluetooth clock as found in the first slot shall be used for the entire packet.

The encryption algorithm E_0 generates a binary keystream, K_{cipher}, which shall be modulo-2 added to the data to be encrypted. The cipher is symmetric; decryption shall be performed in exactly the same way using the same key as used for encryption.

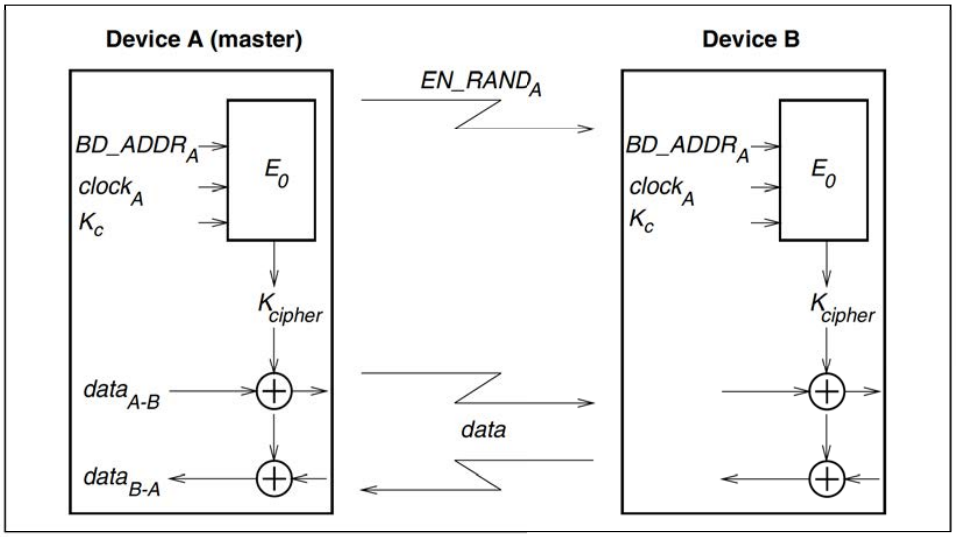


Figure 4.2: Functional description of the encryption procedure

4.4 ENCRYPTION ALGORITHM

The system uses linear feedback shift registers (LFSRs) whose output is combined by a simple finite state machine (called the summation combiner) with 16 states. The output of this state machine is the key stream sequence, or, during initialization phase, the randomized initial start value. The algorithm uses an encryption key K_C, a 48-bit Bluetooth address, the master clock bits $CLK_{26\text{-}1}$, and a 128-bit RAND value. Figure 4.3 on page 791 shows the setup.

There are four LFSRs ($LFSR_1$,...,$LFSR_4$) of lengths $L_1 = 25$, $L_2 = 31$, $L_3 = 33$, and, $L_4 = 39$, with feedback polynomials as specified in Table 4.2 on page 791. The total length of the registers is 128. These polynomials are all primitive. The Hamming weight of all the feedback polynomials is chosen to be five – a reasonable trade-off between reducing the number of required XOR gates in the hardware implementation and obtaining good statistical properties of the generated sequences.

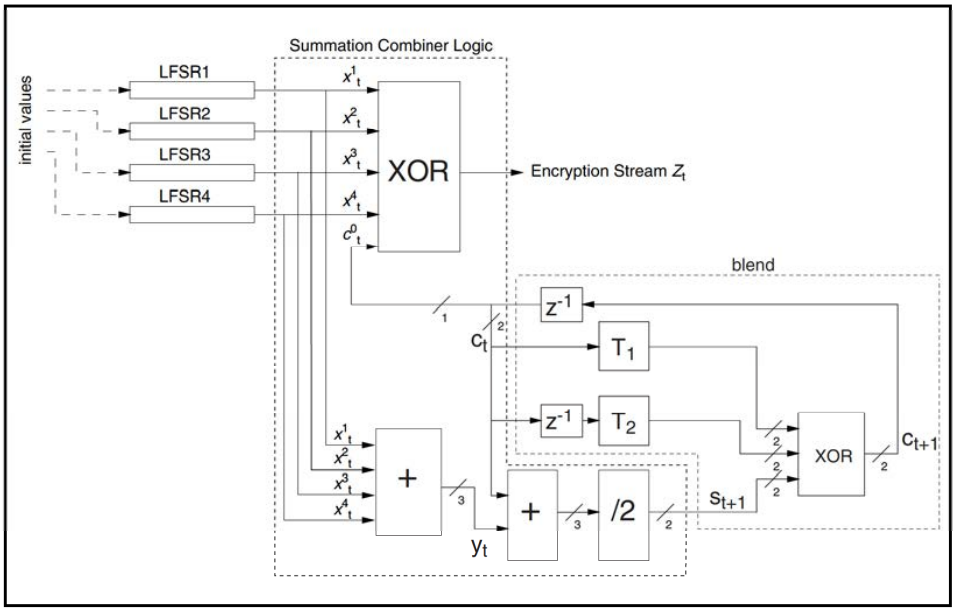


Figure 4.3: Concept of the encryption engine.

i	L_i	**feedback** $f_i(t)$	**weight**
1	25	$t^{25} + t^{20} + t^{12} + t^8 + 1$	5
2	31	$t^{31} + t^{24} + t^{16} + t^{12} + 1$	5
3	33	$t^{33} + t^{28} + t^{24} + t^4 + 1$	5
4	39	$t^{39} + t^{36} + t^{28} + t^4 + 1$	5

Table 4.2: The four primitive feedback polynomials.

Let x_t^i denote the t^{th} symbol of LFSR_i. The value y_t is derived from the four-tuple $x_t^1, \ldots, x_t^4$ using the following equation:

$$y_t = \sum_{i=1}^{4} x_t^i, \qquad \text{(EQ 6)}$$

where the sum is over the integers. Thus y_t can take the values 0,1,2,3, or 4. The output of the summation generator is obtained by the following equations:

$$z_t = x_t^1 \oplus x_t^2 \oplus x_t^3 \oplus x_t^4 \oplus c_t^0 \in \{0, 1\}, \qquad \text{(EQ 7)}$$

$$s_{t+1} = (s^1_{t+1}, s^0_{t+1}) = \left\lfloor \frac{y_t + c_t}{2} \right\rfloor \in \{0, 1, 2, 3\}, \quad \text{(EQ 8)}$$

$$c_{t+1} = (c^1_{t+1}, c^0_{t+1}) = s_{t+1} \oplus T_1[c_t] \oplus T_2[c_{t-1}], \quad \text{(EQ 9)}$$

where $T_1[.]$ and $T_2[.]$ are two different linear bijections over GF(4). Suppose GF(4) is generated by the irreducible polynomial $x^2 + x + 1$, and let α be a zero of this polynomial in GF(4). The mappings T_1 and T_2 are now defined as:

$$T_1: \mathrm{GF}(4) \to \mathrm{GF}(4)$$
$$x \mapsto x$$
$$T_2: \mathrm{GF}(4) \to \mathrm{GF}(4)$$
$$x \mapsto (\alpha + 1)x.$$

The elements of GF(4) can be written as binary vectors. This is summarized in Table 4.3.

x	$T_1[x]$	$T_2[x]$
00	00	00
01	01	11
10	10	01
11	11	10

Table 4.3: The mappings T_1 and T_2.

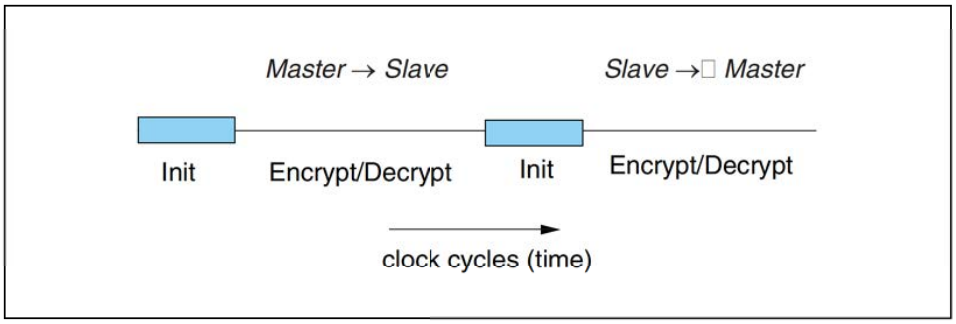
Since the mappings are linear, they can be implemented using XOR gates; i.e.

$$T_1: \quad (x_1, x_0) \mapsto (x_1, x_0),$$
$$T_2: \quad (x_1, x_0) \mapsto (x_0, x_1 \oplus x_0).$$

4.4.1 The operation of the cipher

Figure 4.4 on page 792 gives an overview of the operation in time. The encryption algorithm shall run through the initialization phase before the start of transmission or reception of a new packet. Thus, for multislot packets the cipher is initialized using the clock value of the first slot in the multislot sequence.

Figure 4.4: Overview of the operation of the encryption engine. Between each start of a packet (TX or RX), the LFSRs are re-initialized.



4.5 LFSR INITIALIZATION

The key stream generator is loaded with an initial value for the four LFSRs (in total 128 bits) and the 4 bits that specify the values of c_0 and c_{-1}. The 132 bit initial value is derived from four inputs by using the key stream generator. The input parameters are the key K_C, a 128-bit random number RAND, a 48-bit Bluetooth device address, and the 26 master clock bits $CLK_{26\text{-}1}$.

The effective length of the encryption key may vary between 8 and 128 bits. Note that the actual key length as obtained from E_3 is 128 bits. Then, within E_0, the key length may be reduced by a modulo operation between K_C and a polynomial of desired degree. After reduction, the result is encoded with a block code in order to distribute the starting states more uniformly. The operation shall be as defined in (EQ 10) on page 794.

When the encryption key has been created the LFSRs are loaded with their initial values. Then, 200 stream cipher bits are created by operating the generator. Of these bits, the last 128 are fed back into the key stream generator as an initial value of the four LFSRs. The values of c_t and c_{t-1} are kept. From this point on, when clocked the generator produces the encryption (decryption) sequence which is bitwise XORed to the transmitted (received) payload data.

In the following, octet i of a binary sequence X is $X[i]$. Bit 0 of X is the LSB. Then, the LSB of $X[i]$ corresponds to bit $8i$ of the sequence X, the MSB of $X[i]$ is bit $8i+7$ of X. For instance, bit 24 of the Bluetooth device address is the LSB of BD_ADDR[3].

The details of the initialization shall be as follows:

1. Create the encryption key to use from the 128-bit secret key K_C and the 128-bit publicly known EN_RAND. Let L, $1 \le L \le 16$, be the

effective key length in number of octets. The resulting encryption key is K'_C:

$$K'_C(x) = g_2^{(L)}(x)(K_C(x) \bmod g_1^{(L)}(x)), \qquad \text{(EQ 10)}$$

where $\deg(g_1^{(L)}(x)) = 8L$ and $\deg(g_2^{(L)}(x)) \leq 128 - 8L$. The polynomials are defined in Table 4.4.

2. Shift the 3 inputs K'_C, the Bluetooth device address, the clock, and the six-bit constant 111001 into the LFSRs. In total, 208 bits are shifted in:

   a) Open all switches shown in Figure 4.5 on page 795;

   b) Arrange inputs bits as shown in Figure 4.5; Set the content of all shift register elements to zero. Set $t = 0$.

   c) Start shifting bits into the LFSRs. The rightmost bit at each level of Figure 4.5 is the first bit to enter the corresponding LFSR.

   d) When the first input bit at level i reaches the rightmost position of $LFSR_i$, close the switch of this LFSR.

   e) At $t = 39$ (when the switch of $LFSR_4$ is closed), reset both blend registers $c_{39} = c_{39-1} = 0$; Up to this point, the content of c_t and c_{t-1} has been of no concern. However, their content will now be used in computing the output sequence.

   f) From now on output symbols are generated. The remaining input bits are continuously shifted into their corresponding shift registers. When the last bit has been shifted in, the shift register is clocked with input = 0;

      Note: When finished, $LFSR_1$ has effectively clocked 30 times with feedback closed, $LFSR_2$ has clocked 24 times, $LFSR_3$ has clocked 22 times, and $LFSR_4$ has effectively clocked 16 times with feedback closed.

3. To mix initial data, continue to clock until 200 symbols have been produced with all switches closed ($t = 239$);

4. Keep blend registers c_t and c_{t-1}, make a parallel load of the last 128 generated bits into the LFSRs according to Figure 4.6 at $t = 240$;

After the parallel load in item 4, the blend register contents shall be updated for each subsequent clock.

L	deg	$g_1^{(L)}$	deg	$g_2^{(L)}$
1	[8]	00000000 00000000 00000000 0000011d	[119]	00e275a0 abd218d4 cf928b9b bf6cb08f
2	[16]	00000000 00000000 00000000 0001003f	[112]	0001e3f6 3d7659b3 7f18c258 cff6efef
3	[24]	00000000 00000000 00000000 010000db	[104]	000001be f66c6c3a b1030a5a 1919808b
4	[32]	00000000 00000000 00000001 000000af	[96]	00000001 6ab89969 de17467f d3736ad9

Table 4.4: Polynomials used when creating K'_C. [1]

L	deg	$g_1^{(L)}$	deg	$g_2^{(L)}$
5	[40]	00000000 00000000 00000100 00000039	[88]	00000000 01630632 91da50ec 55715247
6	[48]	00000000 00000000 00010000 00000291	[77]	00000000 00002c93 52aa6cc0 54468311
7	[56]	00000000 00000000 01000000 00000095	[71]	00000000 000000b3 f7fffce2 79f3a073
8	[64]	00000000 00000001 00000000 0000001b	[63]	00000000 00000000 a1ab815b c7ec8025
9	[72]	00000000 00000100 00000000 00000609	[49]	00000000 00000000 0002c980 11d8b04d
10	[80]	00000000 00010000 00000000 00000215	[42]	00000000 00000000 0000058e 24f9a4bb
11	[88]	00000000 01000000 00000000 0000013b	[35]	00000000 00000000 0000000c a76024d7
12	[96]	00000001 00000000 00000000 000000dd	[28]	00000000 00000000 00000000 1c9c26b9
13	[104]	00000100 00000000 00000000 0000049d	[21]	00000000 00000000 00000000 0026d9e3
14	[112]	00010000 00000000 00000000 0000014f	[14]	00000000 00000000 00000000 00004377
15	[120]	01000000 00000000 00000000 000000e7	[7]	00000000 00000000 00000000 00000089
16	[128]	1 00000000 00000000 00000000 00000000	[0]	00000000 00000000 00000000 00000001

Table 4.4: Polynomials used when creating K'_c. . [1]

1. All polynomials are in hexadecimal notation. The LSB is in the rightmost position.

In Figure 4.5, all bits are shifted into the LFSRs, starting with the least significant bit (LSB). For instance, from the third octet of the address, BD_ADDR[2], first BD_ADDR_{16} is entered, followed by BD_ADDR_{17}, etc. Furthermore, CL_0 corresponds to CLK_1,..., CL_{25} corresponds to CLK_{26}.

Note that the output symbols x_t^i, $i = 1, \ldots, 4$ are taken from the positions 24, 24, 32, and 32 for $LFSR_1$, $LFSR_2$,$LFSR_3$, and $LFSR_4$, respectively (counting the leftmost position as number 1).

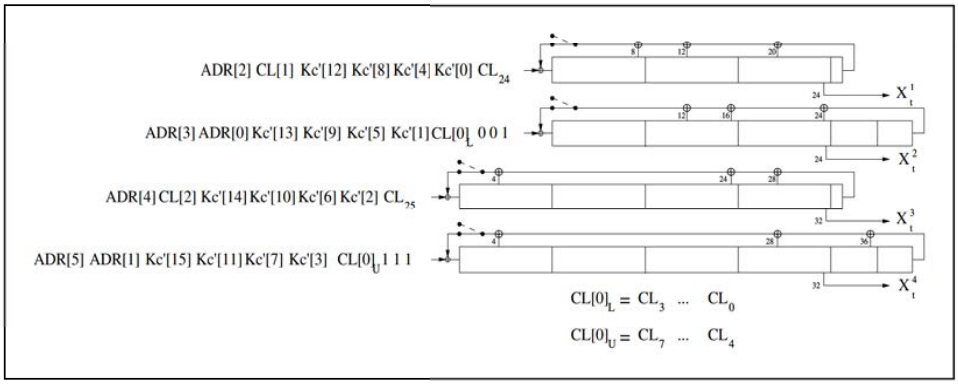


Figure 4.5: Arranging the input to the LFSRs.

In Figure 4.6, the 128 binary output symbols Z_0,..., Z_{127} are arranged in octets denoted Z[0],..., Z[15]. The LSB of Z[0] corresponds to the first of these symbols, the MSB of Z[15] is the last output from the generator. These bits shall be loaded into the LFSRs according to the figure. It is a parallel load and no update of the blend registers is done. The first output symbol is generated at the same time. The octets shall be written into the registers with the LSB in the leftmost position (i.e. the opposite of before). For example, Z_{24} is loaded into position 1 of $LFSR_4$.

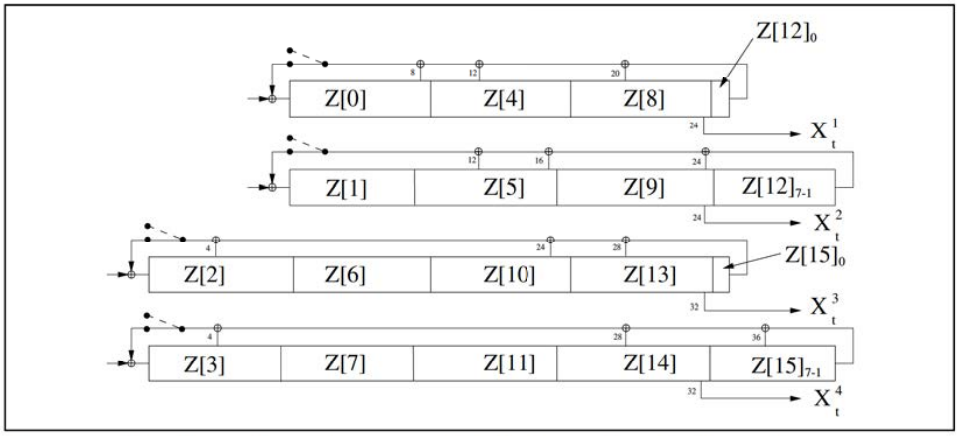


Figure 4.6: Distribution of the 128 last generated output symbols within the LFSRs.

4.6 KEY STREAM SEQUENCE

When the initialization is finished, the output from the summation combiner is used for encryption/decryption. The first bit to use shall be the one produced at the parallel load, i.e. at $t = 240$. The circuit shall be run for the entire length of the current payload. Then, before the reverse direction is started, the entire initialization process shall be repeated with updated values on the input parameters.

Sample data of the encryption output sequence can be found in [Part G] Section 1 on page 673. A necessary, but not sufficient, condition for all Bluetooth compliant implementations of encryption is to produce these encryption streams for identical initialization values.

5 AUTHENTICATION

Authentication uses a challenge-response scheme in which a claimant's knowledge of a secret key is checked through a 2-move protocol using symmetric secret keys. The latter implies that a correct claimant/verifier pair share the same secret key, for example K. In the challenge-response scheme the verifier challenges the claimant to authenticate a random input (the challenge), denoted by AU_RAND_A, with an authentication code, denoted by E_1, and return the result SRES to the verifier, see Figure 5.1 on page 797. This figure also shows that the input to E_1 consists of the tuple AU_RAND_A and the Bluetooth device address (BD_ADDR) of the claimant. The use of this address prevents a simple reflection attack[1]. The secret K shared by devices A and B is the current link key.

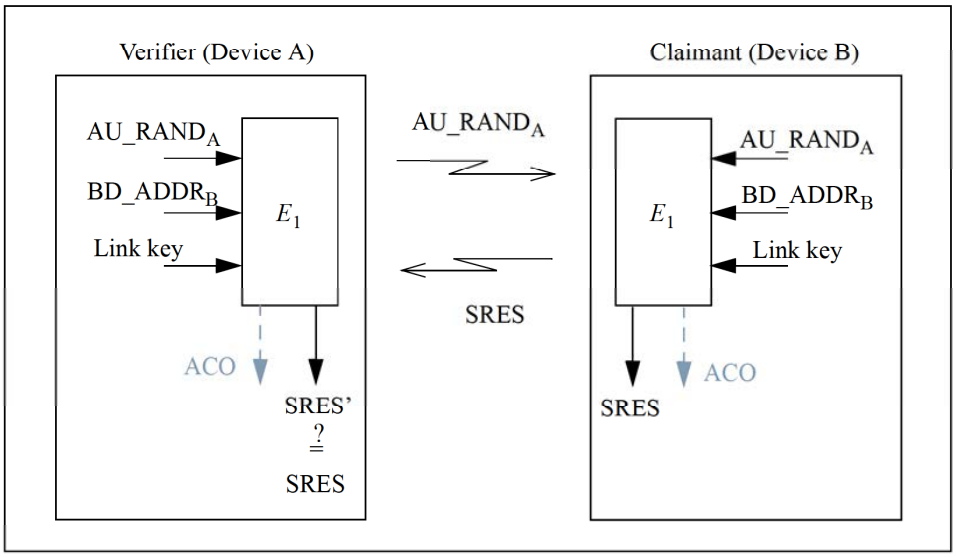


Figure 5.1: Challenge-response for the Bluetooth.

The challenge-response scheme for symmetric keys is depicted in Figure 5.2 on page 798.

1. The reflection attack actually forms no threat because all service requests are dealt with on a FIFO bases. When preemption is introduced, this attack is potentially dangerous.

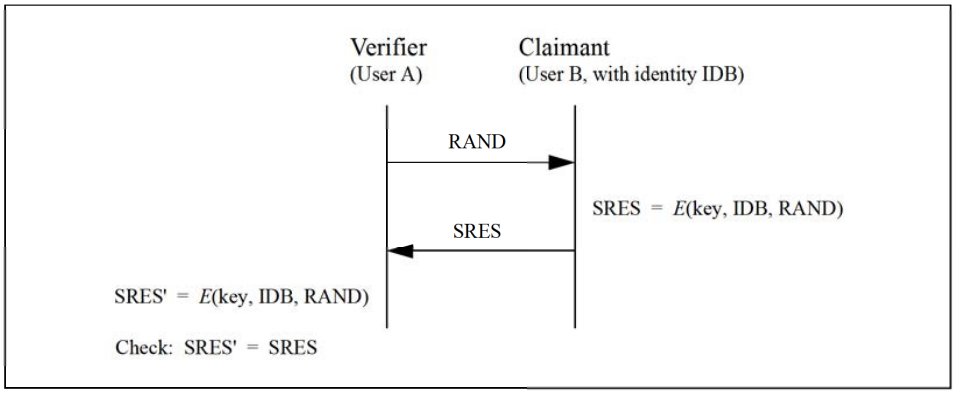


Figure 5.2: Challenge-response for symmetric key systems.

The verifier is not required to be the master. The application indicates which device has to be authenticated. Some applications only require a one-way authentication. However, some peer-to-peer communications, should use a mutual authentication in which each device is subsequently the challenger (verifier) in two authentication procedures. The LM shall process authentication preferences from the application to determine in which direction(s) the authentication(s) takes place. For mutual authentication with the devices of Figure 5.1 on page 797, after device A has successfully authenticated device B, device B could authenticate device A by sending an AU_RAND_B (different from the AU_RAND_A that device A issued) to device A, and deriving the SRES and SRES' from the new AU_RAND_B, the address of device A, and the link key K_{AB}.

If an authentication is successful the value of ACO as produced by E_1 shall be retained.

5.1 REPEATED ATTEMPTS

When the authentication attempt fails, a waiting interval shall pass before the verifier will initiate a new authentication attempt to the same claimant, or before it will respond to an authentication attempt initiated by a device claiming the same identity as the failed device. For each subsequent authentication failure with the same Bluetooth device address, the waiting interval shall be increased exponentially. That is, after each failure, the waiting interval before a new attempt can be made, could be for example, twice as long as the waiting interval prior to the previous attempt[1]. The waiting interval shall be limited to a maximum. The maximum waiting interval depends on the implementation. The waiting time shall exponentially decrease to a minimum when no new failed attempts are made during a certain time period. This procedure prevents an intruder from repeating the authentication procedure with a large number of different keys.

To make the system less vulnerable to denial-of-service attacks, the devices should keep a list of individual waiting intervals for each device it has established contact with. The size of this list may be restricted to only contain the N devices with which the most recent contacts have been made. The number N may vary for different devices depending on available memory size and user environment.

1. Another appropriate value larger than 1 may be used.

6 THE AUTHENTICATION AND KEY-GENERATING FUNCTIONS

This section describes the algorithms used for authentication and key generation.

6.1 THE AUTHENTICATION FUNCTION E_1

The authentication function E_1 is a computationally secure authentication code. E_1 uses the encryption function SAFER+. The algorithm is an enhanced version of an existing 64-bit block cipher SAFER-SK128, and it is freely available. In the following discussion, the block cipher will be denoted as the function A_r which maps using a 128-bit key, a 128-bit input to a 128-bit output, i.e.

$$A_r\colon \{0,1\}^{128} \times \{0,1\}^{128} \rightarrow \{0,1\}^{128}$$
$$(k \times x) \mapsto t. \qquad \text{(EQ 11)}$$

The details of A_r are given in the next section. The function E_1 is constructed using A_r as follows

$$E_1\colon \{0,1\}^{128} \times \{0,1\}^{128} \times \{0,1\}^{48} \rightarrow \{0,1\}^{32} \times \{0,1\}^{96}$$
$$(K, \text{RAND}, \text{address}) \mapsto (\text{SRES}, \text{ACO}), \qquad \text{(EQ 12)}$$

where $\text{SRES} = Hash(K, \text{RAND}, \text{address}, 6)[0, \ldots, 3]$, where $Hash$ is a keyed hash function defined as[1],

$$Hash\colon \{0,1\}^{128} \times \{0,1\}^{128} \times \{0,1\}^{8 \times L} \times \{6,12\} \rightarrow \{0,1\}^{128}$$
$$(K, I_1, I_2, L) \mapsto A'_r([\tilde{K}], [E(I_2, L) +_{16} (A_r(K, I_1) \oplus_{16} I_1)]), \qquad \text{(EQ 13)}$$

and where

$$E\colon \{0,1\}^{8 \times L} \times \{6,12\} \rightarrow \{0,1\}^{8 \times 16}$$
$$(X[0, \ldots, L-1], L) \mapsto (X[i(\text{mod L})] \quad \text{for } i = 0 \ldots 15), \qquad \text{(EQ 14)}$$

is an expansion of the L octet word X into a 128-bit word. The function A_r is evaluated twice for each evaluation of E_1. The key $\tilde{K}$ for the second use of A_r (actually A'_r) is offset from K as follows[2]

1. The operator $+_{16}$ denotes bytewise addition mod 256 of the 16 octets, and the operator $\oplus_{16}$ denotes bytewise XORing of the 16 octets.
2. The constants are the largest primes below 257 for which 10 is a primitive root.

$$\begin{aligned}
\tilde{K}[0] &= (K[0] + 233) \mod 256, & \tilde{K}[1] &= K[1] \oplus 229, \\
\tilde{K}[2] &= (K[2] + 223) \mod 256, & \tilde{K}[3] &= K[3] \oplus 193, \\
\tilde{K}[4] &= (K[4] + 179) \mod 256, & \tilde{K}[5] &= K[5] \oplus 167, \\
\tilde{K}[6] &= (K[6] + 149) \mod 256, & \tilde{K}[7] &= K[7] \oplus 131, \\
\tilde{K}\{8\} &= K[8] \oplus 233, & \tilde{K}[9] &= (K[9] + 229) \mod 256, \\
\tilde{K}[10] &= K[10] \oplus 223, & \tilde{K}[11] &= (K[11] + 193) \mod 256, \\
\tilde{K}[12] &= K[12] \oplus 179, & \tilde{K}[13] &= (K[13] + 167) \mod 256, \\
\tilde{K}[14] &= K[14] \oplus 149, & \tilde{K}[15] &= (K[15] + 131) \mod 256.
\end{aligned} \qquad \text{(EQ 15)}$$

A data flowchart of the computation of E_1 is shown in Figure 6.1 on page 802. E_1 is also used to deliver the parameter ACO (Authenticated Ciphering Offset) that is used in the generation of the ciphering key by E_3, see equations (EQ 3) on page 783 and (EQ 23) on page 807. The value of ACO is formed by the octets 4 through 15 of the output of the hash function defined in (EQ 13) on page 801, i.e.

$$\text{ACO} = Hash(K, \text{RAND}, \text{address}, 6)[4, \ldots, 15]. \qquad \text{(EQ 16)}$$

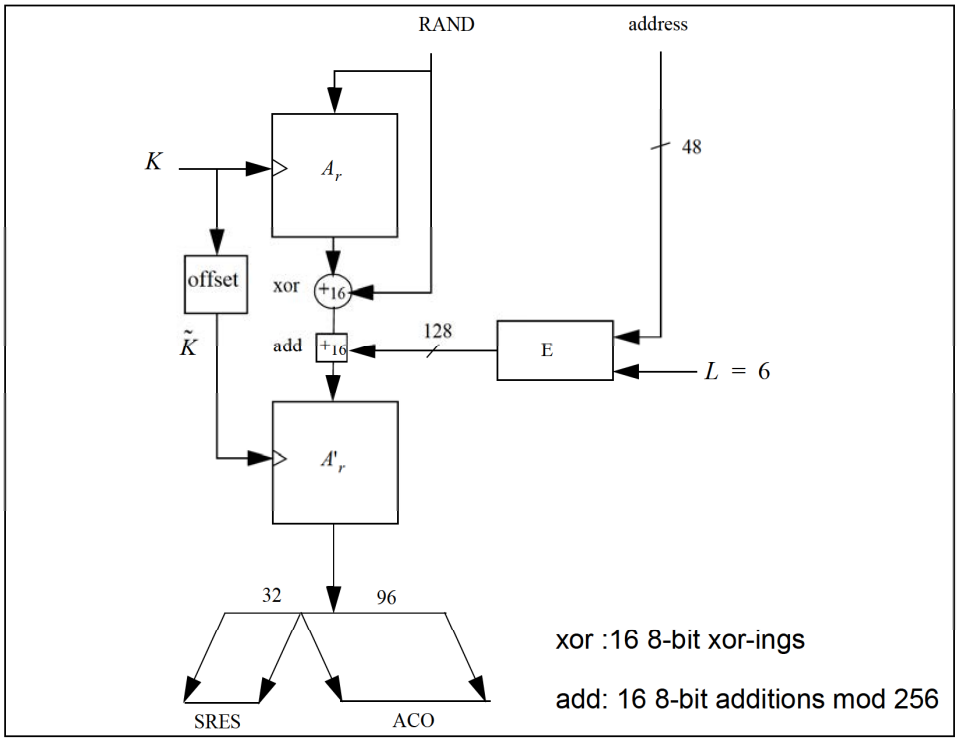


Figure 6.1: Flow of data for the computation of E_1.

6.2 THE FUNCTIONS A_r AND A'_r

The function A_r is identical to SAFER+. It consists of a set of 8 layers, (each layer is called a round) and a parallel mechanism for generating the sub keys $K_p[j]$, $p = 1, 2, \ldots, 17$, which are the round keys to be used in each round. The function will produce a 128-bit result from a 128-bit random input string and a 128-bit key. Besides the function A_r, a slightly modified version referred to as A'_r is used in which the input of round 1 is added to the input of round 3. This is done to make the modified version non-invertible and prevents the use of A'_r (especially in E_{2x}) as an encryption function. See Figure 6.2 on page 804 for details.

6.2.1 The round computations

The computations in each round are a composition of encryption with a round key, substitution, encryption with the next round key, and, finally, a Pseudo Hadamard Transform (PHT). The computations in a round shall be as shown in Figure 6.2 on page 804. The sub keys for round $r, r = 1, 2, \ldots, 8$ are denoted $K_{2r-1}[j]$, $K_{2r}[j]$, $j = 0, 1, \ldots, 15$. After the last round $K_{17}[j]$ is applied identically to all previous odd numbered keys.

6.2.2 The substitution boxes “e” and “l”

In Figure 6.2 on page 804 two boxes are shown, marked “e” and “l”. These boxes implement the same substitutions as are used in SAFER+; i.e. they implement

$$\begin{aligned} e, l &: \{0, \ldots, 255\} \rightarrow \{0, \ldots, 255\}, \\ e &: i \mapsto (45^i \pmod{257}) \pmod{256}, \\ l &: i \mapsto j \text{ s.t. } i = e(j). \end{aligned}$$

Their role, as in the SAFER+ algorithm, is to introduce non-linearity.

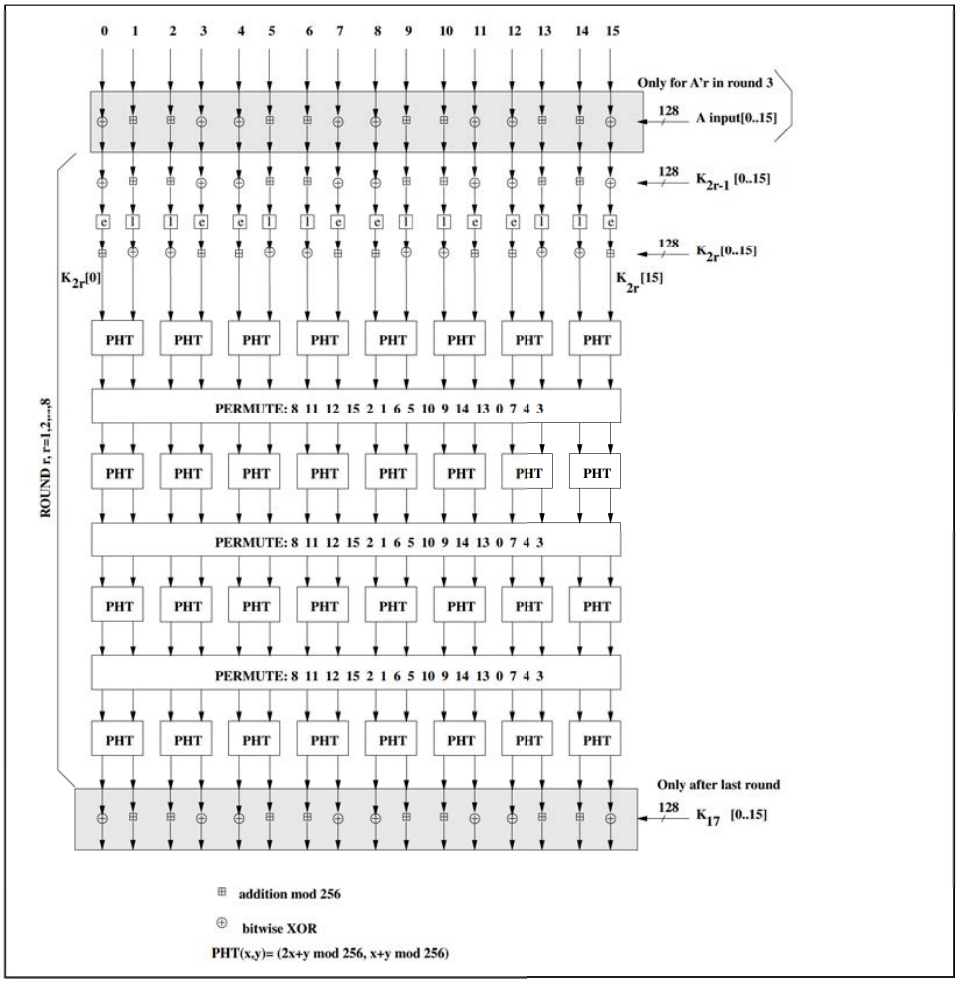


Figure 6.2: One round in A_r and A'_r. The permutation boxes show how input byte indices are mapped onto output byte indices. Thus, position 8 is mapped on position 0 (leftmost), position 11 is mapped on position 1, etc.

6.2.3 Key scheduling

In each round, 2 batches of 16 octet-wide keys are needed. These round keys are derived as specified by the key scheduling in SAFER+. Figure 6.3 on page 805 gives an overview of how the round keys $K_p[j]$ are determined. The bias vectors B_2, B_3, ..., B_{17} shall be computed according to following equation:

$$B_p[i] = ((45^{(45^{17p+i+1} \bmod 257)} \bmod 257) \bmod 256), \text{ for } i = 0, \ldots, 15. \qquad \text{(EQ 17)}$$

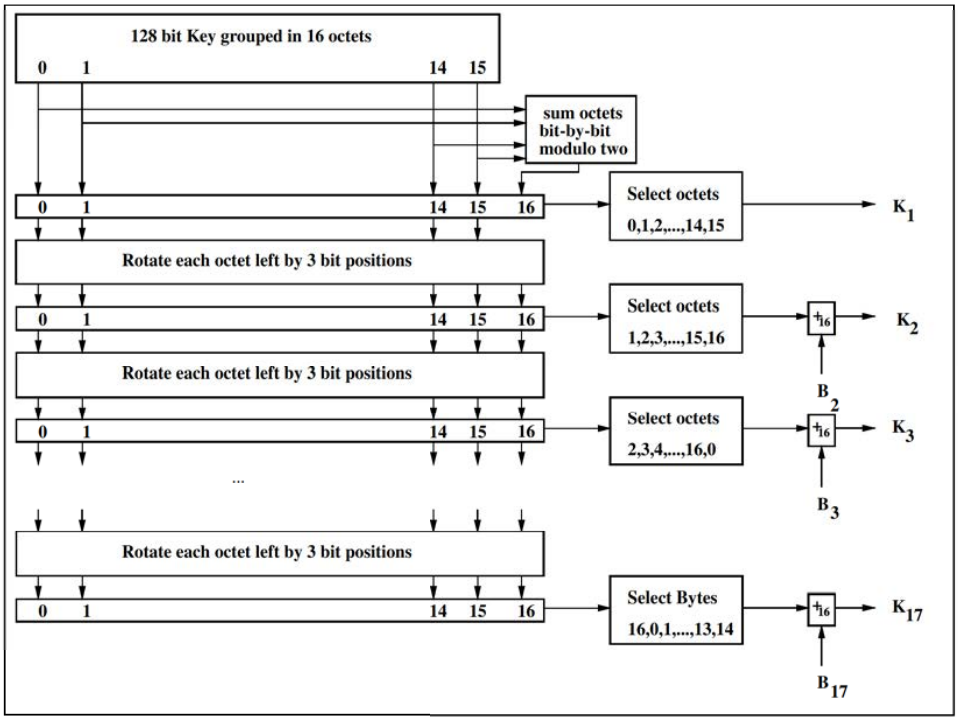


Figure 6.3: Key scheduling in A_r.

6.3 E_2-KEY GENERATION FUNCTION FOR AUTHENTICATION

The key used for authentication shall be derived through the procedure that is shown in Figure 6.4 on page 807. The figure shows two modes of operation for the algorithm. In the first mode, E_{21} produces a 128-bit kink key, K, using a 128-bit RAND value and a 48-bit address. This mode shall be utilized when creating unit keys and combination keys. In the second mode, E_{22} produces a 128-bit link key, K, using a 128-bit RAND value and an L octet user PIN. The second mode shall be used to create the initialization key, and also when a master key is to be generated.

When the initialization key is generated, the PIN is augmented with the BD_ADDR, see Section 3.2.1 on page 780 for which address to use. The augmentation shall always start with the least significant octet of the address immediately following the most significant octet of the PIN. Since the maximum length of the PIN used in the algorithm cannot exceed 16 octets, it is possible that not all octets of BD_ADDR will be used.

This key generating algorithm again exploits the cryptographic function. E_2 for mode 1 (denoted E_{21}) is computed according to following equations:

$$E_{21}: \{0,1\}^{128} \times \{0,1\}^{48} \rightarrow \{0,1\}^{128}$$
$$(\text{RAND}, \text{address}) \mapsto A'_r(X, Y) \qquad \text{(EQ 18)}$$

where (for mode 1)

$$\begin{cases} X = \text{RAND}[0 \ldots 14] \cup (\text{RAND}[15] \oplus 6) \\ Y = \bigcup_{i=0}^{15} \text{address}[i \pmod 6] \end{cases} \qquad \text{(EQ 19)}$$

Let L be the number of octets in the user PIN. The augmenting is defined by

$$\text{PIN}' = \begin{cases} \text{PIN}[0 \ldots L-1] \cup \text{BD\_ADDR}[0 \ldots \min\{5, 15-L\}], & L < 16, \\ \text{PIN}[0 \ldots L-1], & L = 16, \end{cases} \qquad \text{(EQ 20)}$$

Then, in mode 2, E_2 (denoted E_{22}) is

$$E_{22}: \{0,1\}^{8L'} \times \{0,1\}^{128} \times \{1, 2, \ldots, 16\} \rightarrow \{0,1\}^{128}$$
$$(\text{PIN}', \text{RAND}, \text{L}') \mapsto A'_r(X, Y) \qquad \text{(EQ 21)}$$

where

$$\begin{cases} X = \bigcup_{i=0}^{15} \text{PIN}'[i \pmod{L'}], \\ Y = \text{RAND}[0 \ldots 14] \cup (\text{RAND}[15] \oplus L'), \end{cases} \qquad \text{(EQ 22)}$$

and $L' = \min\{16, L+6\}$ is the number of octets in PIN'.

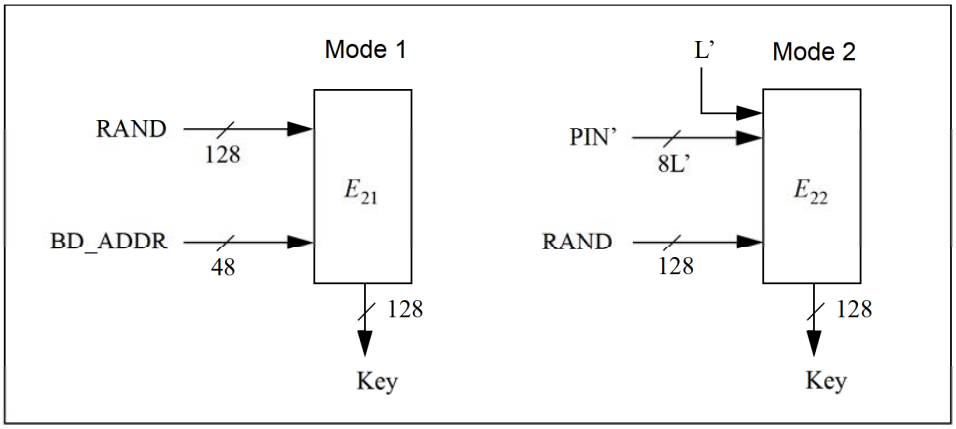


Figure 6.4: Key generating algorithm E_2 and its two modes. Mode 1 is used for unit and combination keys, while mode 2 is used for K_{init} and K_{master}.

6.4 E_3-KEY GENERATION FUNCTION FOR ENCRYPTION

The ciphering key K_C used by E_0 shall be generated by E_3. The function E_3 is constructed using A'_r as follows

$$E_3: \{0,1\}^{128} \times \{0,1\}^{128} \times \{0,1\}^{96} \rightarrow \{0,1\}^{128}$$
$$(K, \text{RAND}, \text{COF}) \mapsto Hash(K, \text{RAND}, \text{COF}, 12) \qquad \text{(EQ 23)}$$

where $Hash$ is the hash function as defined by (EQ 13) on page 801. The key length produced is 128 bits. However, before use within E_0, the encryption key K_C is shortened to the correct encryption key length, as described in Section 4.5 on page 793. A block scheme of E_3 is depicted in Figure 6.5.

The value of COF is determined as specified by equation (EQ 3) on page 783.

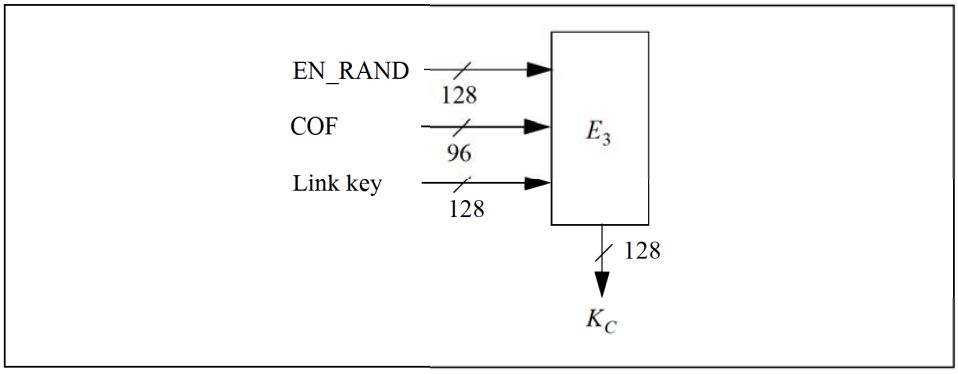


Figure 6.5: Generation of the encryption key.

7 LIST OF FIGURES

Figure 3.1: Generation of unit key. When the unit key has been exchanged, the initialization key is discarded in both devices. 777
Figure 3.2: Generating a combination key. The old link key (K) is discarded after the exchange of a new combination key has succeeded 778
Figure 3.3: Master link key distribution and computation of the corresponding encryption key. 781
Figure 4.1: Stream ciphering for Bluetooth with E0. 783
Figure 4.2: Functional description of the encryption procedure 786
Figure 4.3: Concept of the encryption engine. 787
Figure 4.4: Overview of the operation of the encryption engine. Between each start of a packet (TX or RX), the LFSRs are re-initialized. 788
Figure 4.5: Arranging the input to the LFSRs. 791
Figure 4.6: Distribution of the 128 last generated output symbols within the LFSRs. 792
Figure 5.1: Challenge-response for the Bluetooth. 793
Figure 5.2: Challenge-response for symmetric key systems. 794
Figure 6.1: Flow of data for the computation of E_1. 798
Figure 6.2: One round in and . The permutation boxes show how input byte indices are mapped onto output byte indices. Thus, position 8 is mapped on position 0 (leftmost), position 11 is mapped on position 1, etc. 800
Figure 6.3: Key scheduling in A_r. 801
Figure 6.4: Key generating algorithm E_2 and its two modes. Mode 1 is used for unit and combination keys, while mode 2 is used for K_{init} and K_{master}. 803
Figure 6.5: Generation of the encryption key. 803

8 LIST OF TABLES

Table 1.1: Entities used in authentication and encryption procedures.....769
Table 4.1: Possible encryption modes for a slave in possession of a master key.785
Table 4.2: The four primitive feedback polynomials.....787
Table 4.3: The mappings $T1$ and $T2$. 788
Table 4.4: Polynomials used when creating $K'c$. 790





Specification Volume 4

Specification of the Bluetooth System

Wireless connections made easy

Core System Package

[Host volume]

Covered Core Package version:
2.0 + EDR
Current Master TOC issued:
4 November 2004

Bluetooth®

Revision History

The Revision History is shown in the “Appendix” on page 51[vol. 0].

Contributors

The persons who contributed to this specification are listed in the “Appendix” on page 51[vol. 0].

Web Site

This specification can also be found on the official Bluetooth web site:
http://www.bluetooth.com

Disclaimer and Copyright Notice

The copyright in these specifications is owned by the Promoter Members of Bluetooth SIG, Inc. (“Bluetooth SIG”). Use of these specifications and any related intellectual property (collectively, the “Specification”), is governed by the Promoters Membership Agreement among the Promoter Members and Bluetooth SIG (the “Promoters Agreement”), certain membership agreements between Bluetooth SIG and its Adopter and Associate Members (the “Membership Agreements”) and the Bluetooth Specification Early Adopters Agreements (“1.2 Early Adopters Agreements”) among Early Adopter members of the unincorporated Bluetooth special interest group and the Promoter Members (the “Early Adopters Agreement”). Certain rights and obligations of the Promoter Members under the Early Adopters Agreements have been assigned to Bluetooth SIG by the Promoter Members.

Use of the Specification by anyone who is not a member of Bluetooth SIG or a party to an Early Adopters Agreement (each such person or party, a “Member”), is prohibited. The legal rights and obligations of each Member are governed by their applicable Membership Agreement, Early Adopters Agreement or Promoters Agreement. No license, express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

Any use of the Specification not in compliance with the terms of the applicable Membership Agreement, Early Adopters Agreement or Promoters Agreement is prohibited and any such prohibited use may result in termination of the applicable Membership Agreement or Early Adopters Agreement and other liability permitted by the applicable agreement or by applicable law to Bluetooth SIG or any of its members for patent, copyright and/or trademark infringement.



THE SPECIFICATION IS PROVIDED “AS IS” WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, SATISFACTORY QUALITY, OR REASONABLE SKILL OR CARE, OR ANY WARRANTY ARISING OUT OF ANY COURSE OF DEALING, USAGE, TRADE PRACTICE, PROPOSAL, SPECIFICATION OR SAMPLE.

Each Member hereby acknowledges that products equipped with the Bluetooth® technology (“Bluetooth® Products”) may be subject to various regulatory controls under the laws and regulations of various governments worldwide. Such laws and regulatory controls may govern, among other things, the combination, operation, use, implementation and distribution of Bluetooth® Products. Examples of such laws and regulatory controls include, but are not limited to, airline regulatory controls, telecommunications regulations, technology transfer controls and health and safety regulations. Each Member is solely responsible for the compliance by their Bluetooth® Products with any such laws and regulations and for obtaining any and all required authorizations, permits, or licenses for their Bluetooth® Products related to such regulations within the applicable jurisdictions. Each Member acknowledges that nothing in the Specification provides any information or assistance in connection with securing such compliance, authorizations or licenses. NOTHING IN THE SPECIFICATION CREATES ANY WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING SUCH LAWS OR REGULATIONS.

ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHTS OR FOR NONCOMPLIANCE WITH LAWS, RELATING TO USE OF THE SPECIFICATION IS EXPRESSLY DISCLAIMED. BY USE OF THE SPECIFICATION, EACH MEMBER EXPRESSLY WAIVES ANY CLAIM AGAINST BLUETOOTH SIG AND ITS PROMOTER MEMBERS RELATED TO USE OF THE SPECIFICATION.

Bluetooth SIG reserves the right to adopt any changes or alterations to the Specification as it deems necessary or appropriate.

Copyright © 1999, 2000, 2001, 2002, 2003, 2004

Agere Systems, Inc.,
Ericsson Technology Licensing, AB,
IBM Corporation,
Intel Corporation,
Microsoft Corporation,
Motorola, Inc.,
Nokia Corporation,
Toshiba Corporation

*Third-party brands and names are the property of their respective owners.

Part A

Logical Link Control and Adaptation Protocol

1 Introduction 21
- 1.1 L2CAP Features 21
- 1.2 Assumptions 25
- 1.3 Scope 25
- 1.4 Terminology 26

2 General Operation 29
- 2.1 Channel Identifiers 29
- 2.2 Operation Between Devices 29
- 2.3 Operation Between Layers 31
- 2.4 Modes of Operation 31

3 Data Packet Format 33
- 3.1 Connection-oriented Channel in Basic L2CAP Mode 33
- 3.2 Connectionless Data Channel in Basic L2CAP Mode 34
- 3.3 Connection-oriented Channel in Retransmission/Flow Control Modes 35
  - 3.3.1 L2CAP header fields 35
  - 3.3.2 Control field (2 octets) 36
  - 3.3.3 L2CAP SDU length field (2 octets) 38
  - 3.3.4 Information payload field (0 to 65531 octets) 38
  - 3.3.5 Frame check sequence (2 octets) 39
  - 3.3.6 Invalid frame detection 40

4 Signalling Packet Formats 41
- 4.1 Command Reject (code 0x01) 43
- 4.2 Connection Request (code 0x02) 44
- 4.3 Connection Response (code 0x03) 46
- 4.4 Configuration Request (code 0x04) 47
- 4.5 Configuration Response (code 0x05) 50
- 4.6 Disconnection Request (code 0x06) 52
- 4.7 Disconnection Response (code 0x07) 53
- 4.8 Echo Request (code 0x08) 53
- 4.9 Echo Response (code 0x09) 54
- 4.10 Information Request (code 0x0A) 54
- 4.11 Information Response (code 0x0B) 55
- 4.12 Extended Feature Mask 56

5 Configuration Parameter Options 57
- 5.1 Maximum Transmission Unit (MTU) 57
- 5.2 Flush Timeout Option 59
- 5.3 Quality of Service (QoS) Option 60

5.4 Retransmission and Flow Control Option 64
6 State Machine 67
6.1 General rules for the state machine: 67
6.1.1 CLOSED state 68
6.1.2 WAIT_CONNECT_RSP state 69
6.1.3 WAIT_CONNECT state 69
6.1.4 CONFIG state 70
6.1.5 OPEN state 73
6.1.6 WAIT_DISCONNECT state 73
6.2 Timers events 75
6.2.1 RTX 75
6.2.2 ERTX 76
7 General Procedures 79
7.1 Configuration Process 79
7.1.1 Request path 80
7.1.2 Response path 80
7.2 Fragmentation and Recombination 81
7.2.1 Fragmentation of L2CAP PDUs 81
7.2.2 Recombination of L2CAP PDUs 82
7.3 Encapsulation of SDUs 83
7.3.1 Segmentation of L2CAP SDUs 83
7.3.2 Reassembly of L2CAP SDUs 84
7.3.3 Segmentation and fragmentation 84
7.4 Delivery of Erroneous L2CAP SDUs 85
7.5 Operation with Flushing 85
7.6 Connectionless Data Channel 86
8 Procedures for Flow Control and Retransmission 87
8.1 Information Retrieval 87
8.2 Function of PDU Types for Flow Control and Retransmission 87
8.2.1 Information frame (I-frame) 87
8.2.2 Supervisory Frame (S-frame) 87
8.2.2.1 Receiver Ready (RR) 87
8.2.2.2 Reject (REJ) 88
8.3 Variables and Sequence Numbers 89
8.3.1 Sending peer 89
8.3.1.1 Send sequence number TxSeq 89
8.3.1.2 Send state variable NextTXSeq 89
8.3.1.3 Acknowledge state variable ExpectedAckSeq 90
8.3.2 Receiving peer 91
8.3.2.1 Receive sequence number ReqSeq 91

8.3.2.2 Receive state variable, ExpectedTxSeq 91
8.3.2.3 Buffer state variable BufferSeq 91
8.4 Retransmission Mode 93
8.4.1 Transmitting frames 93
8.4.1.1 Last received R was set to zero 93
8.4.1.2 Last received R was set to one 95
8.4.2 Receiving I-frames 95
8.4.3 I-frames pulled by the SDU reassembly function 95
8.4.4 Sending and receiving acknowledgements 96
8.4.4.1 Sending acknowledgements 96
8.4.4.2 Receiving acknowledgements 96
8.4.5 Receiving REJ frames 97
8.4.6 Waiting acknowledgements 97
8.4.7 Exception conditions 97
8.4.7.1 TxSeq Sequence error 97
8.4.7.2 ReqSeq Sequence error 98
8.4.7.3 Timer recovery error 98
8.4.7.4 Invalid frame 98
8.5 Flow Control Mode 99
8.5.1 Transmitting I-frames 99
8.5.2 Receiving I-frames 100
8.5.3 I-frames pulled by the SDU reassembly function 100
8.5.4 Sending and receiving acknowledgements 100
8.5.4.1 Sending acknowledgements 100
8.5.4.2 Receiving acknowledgements 100
8.5.5 Waiting acknowledgements 101
8.5.6 Exception conditions 101
8.5.6.1 TxSeq Sequence error 101
8.5.6.2 ReqSeq Sequence error 102
8.5.6.3 Timer recovery error 102
8.5.6.4 Invalid frame 102
9 List of Figures 103
10 List of Tables 105
11 Appendix 105

Part B

Service Discovery Protocol

1 Introduction 115
1.1 General Description 115
1.2 Motivation 115
1.3 Requirements 115

1.4 Non-requirements and Deferred Requirements 116
1.5 Conventions 116
1.5.1 Bit And Byte Ordering Conventions 116
2 Overview 117
2.1 SDP Client-Server Interaction 117
2.2 Service Record 118
2.3 Service Attribute 120
2.4 Attribute ID 120
2.5 Attribute Value 121
2.6 Service Class 121
2.6.1 A Printer Service Class Example 122
2.7 Searching for Services 123
2.7.1 UUID 123
2.7.2 Service Search Patterns 124
2.8 Browsing for Services 124
2.8.1 Example Service Browsing Hierarchy 125
3 Data Representation 127
3.1 Data Element 127
3.2 Data Element Type Descriptor 127
3.3 Data Element Size Descriptor 128
3.4 Data Element Examples 129
4 Protocol Description 131
4.1 Transfer Byte Order 131
4.2 Protocol Data Unit Format 131
4.3 Partial Responses and Continuation State 133
4.4 Error Handling 133
4.4.1 SDP_ErrorResponse PDU 134
4.5 ServiceSearch Transaction 135
4.5.1 SDP_ServiceSearchRequest PDU 135
4.5.2 SDP_ServiceSearchResponse PDU 136
4.6 ServiceAttribute Transaction 138
4.6.1 SDP_ServiceAttributeRequest PDU 138
4.6.2 SDP_ServiceAttributeResponse PDU 140
4.7 ServiceSearchAttribute Transaction 141
4.7.1 SDP_ServiceSearchAttributeRequest PDU 141
4.7.2 SDP_ServiceSearchAttributeResponse PDU 143
5 Service Attribute Definitions 145
5.1 Universal Attribute Definitions 145
5.1.1 ServiceRecordHandle Attribute 145
5.1.2 ServiceClassIDList Attribute 146

5.1.3 ServiceRecordState Attribute 146
5.1.4 ServiceID Attribute 146
5.1.5 ProtocolDescriptorList Attribute 147
5.1.6 BrowseGroupList Attribute 148
5.1.7 LanguageBaseAttributeIDList Attribute 148
5.1.8 ServiceInfoTimeToLive Attribute 149
5.1.9 ServiceAvailability Attribute 150
5.1.10 BluetoothProfileDescriptorList Attribute 150
5.1.11 DocumentationURL Attribute 151
5.1.12 ClientExecutableURL Attribute 151
5.1.13 IconURL Attribute 152
5.1.14 ServiceName Attribute 152
5.1.15 ServiceDescription Attribute 153
5.1.16 ProviderName Attribute 153
5.1.17 Reserved Universal Attribute IDs 153
5.2 ServiceDiscoveryServer Service Class Attribute Definitions 154
5.2.1 ServiceRecordHandle Attribute 154
5.2.2 ServiceClassIDList Attribute 154
5.2.3 VersionNumberList Attribute 154
5.2.4 ServiceDatabaseState Attribute 155
5.2.5 Reserved Attribute IDs 155
5.3 BrowseGroupDescriptor Service Class Attribute Definitions 155
5.3.1 ServiceClassIDList Attribute 155
5.3.2 GroupID Attribute 156
5.3.3 Reserved Attribute IDs 156
6 Appendix 156

Part C

Generic Access Protocol

1 Introduction 179
1.1 Scope 179
1.2 Symbols and conventions 179
1.2.1 Requirement status symbols 179
1.2.2 Signaling diagram conventions 180
1.2.3 Notation for timers and counters 180
2 Profile overview 181
2.1 Profile stack 181

2.2 Configurations and roles 181
2.3 User requirements and scenarios 182
2.4 Profile fundamentals 183
2.5 Conformance 183

3 User interface aspects 185
3.1 The user interface level 185
3.2 Representation of Bluetooth parameters 185
3.2.1 Bluetooth device address (BD_ADDR) 185
3.2.1.1 Definition 185
3.2.1.2 Term on user interface level 185
3.2.1.3 Representation 185
3.2.2 Bluetooth device name (the user-friendly name) 185
3.2.2.1 Definition 185
3.2.2.2 Term on user interface level 186
3.2.2.3 Representation 186
3.2.3 Bluetooth passkey (Bluetooth PIN) 186
3.2.3.1 Definition 186
3.2.3.2 Terms at user interface level 186
3.2.3.3 Representation 186
3.2.4 Class of Device 187
3.2.4.1 Definition 187
3.2.4.2 Term on user interface level 188
3.2.4.3 Representation 188
3.3 Pairing 188

4 Modes 189
4.1 Discoverability modes 189
4.1.1 Non-discoverable mode 190
4.1.1.1 Definition 190
4.1.1.2 Term on UI-level 190
4.1.2 Limited discoverable mode 190
4.1.2.1 Definition 190
4.1.2.2 Conditions 191
4.1.2.3 Term on UI-level 191
4.1.3 General discoverable mode 191
4.1.3.1 Definition 191
4.1.3.2 Conditions 192
4.1.3.3 Term on UI-level 192
4.2 Connectability modes 193
4.2.1 Non-connectable mode 193
4.2.1.1 Definition 193
4.2.1.2 Term on UI-level 193
4.2.2 Connectable mode 193
4.2.2.1 Definition 193

4.2.2.2 Term on UI-level 194
4.3 Pairing modes 195
4.3.1 Non-pairable mode 195
4.3.1.1 Definition 195
4.3.1.2 Term on UI-level 195
4.3.2 Pairable mode 195
4.3.2.1 Definition 195
4.3.2.2 Term on UI-level 195
5 Security aspects 197
5.1 Authentication 197
5.1.1 Purpose 197
5.1.2 Term on UI level 197
5.1.3 Procedure 198
5.1.4 Conditions 198
5.2 Security modes 198
5.2.1 Security mode 1 (non-secure) 200
5.2.2 Security mode 2 (service level enforced security) 200
5.2.3 Security modes 3 (link level enforced security) 200
6 Idle mode procedures 201
6.1 General inquiry 201
6.1.1 Purpose 201
6.1.2 Term on UI level 201
6.1.3 Description 202
6.1.4 Conditions 202
6.2 Limited inquiry 202
6.2.1 Purpose 202
6.2.2 Term on UI level 203
6.2.3 Description 203
6.2.4 Conditions 203
6.3 Name discovery 204
6.3.1 Purpose 204
6.3.2 Term on UI level 204
6.3.3 Description 204
6.3.3.1 Name request 204
6.3.3.2 Name discovery 204
6.3.4 Conditions 205
6.4 Device discovery 205
6.4.1 Purpose 205
6.4.2 Term on UI level 205

6.4.3 Description 206
6.4.4 Conditions 206
6.5 Bonding 206
6.5.1 Purpose 206
6.5.2 Term on UI level 206
6.5.3 Description 207
6.5.3.1 General bonding 207
6.5.3.2 Dedicated bonding 208
6.5.4 Conditions 208
7 Establishment procedures 209
7.1 Link establishment 209
7.1.1 Purpose 209
7.1.2 Term on UI level 209
7.1.3 Description 210
7.1.3.1 B in security mode 1 or 2 210
7.1.3.2 B in security mode 3 211
7.1.4 Conditions 211
7.2 Channel establishment 212
7.2.1 Purpose 212
7.2.2 Term on UI level 212
7.2.3 Description 212
7.2.3.1 B in security mode 2 213
7.2.3.2 B in security mode 1 or 3 213
7.2.4 Conditions 213
7.3 Connection establishment 214
7.3.1 Purpose 214
7.3.2 Term on UI level 214
7.3.3 Description 214
7.3.3.1 B in security mode 2 214
7.3.3.2 B in security mode 1 or 3 215
7.3.4 Conditions 215
7.4 Establishment of additional connection 215
8 Definitions 217
8.1 General definitions 217
8.2 Connection-related definitions 217
8.3 Device-related definitions 218
8.4 Procedure-related definitions 219
8.5 Security-related definitions 219
9 Appendix A (Normative): Timers and constants 221

10 Appendix B (Informative): Information flows of related procedures. 223
10.1 Imp-authentication....223
10.2 Imp-pairing....224
10.3 Service discovery....225
11 References....227

Part D

Test Support

1 Test Methodology....231
1.1 Test Scenarios....231
1.1.1 Test setup....231
1.1.2 Transmitter Test....232
1.1.2.1 Packet Format....233
1.1.2.2 Pseudorandom Sequence....234
1.1.2.3 Control of Transmit Parameters....235
1.1.2.4 Power Control....235
1.1.2.5 Switch Between Different Frequency Settings....235
1.1.2.6 Adaptive Frequency Hopping....236
1.1.3 LoopBack test....237
1.1.4 Pause test....240
1.2 References....240
2 Test Control Interface (TCI)....241
2.1 Introduction....241
2.1.1 Terms used....241
2.1.2 Usage of the interface....241
2.2 TCI Configurations....242
2.2.1 Bluetooth RF requirements....242
2.2.1.1 Required interfaces....242
2.2.2 Bluetooth protocol requirements....243
2.2.2.1 Required interfaces....243
2.2.3 Bluetooth profile requirements....244
2.2.3.1 Required interfaces....244
2.3 TCI Configuration and Usage....245
2.3.1 Transport layers....245
2.3.1.1 Physical bearer....245
2.3.1.2 Software bearer....245
2.3.2 Baseband and link manager qualification....246
2.3.3 HCI qualification....248



LOGICAL LINK CONTROL AND ADAPTATION PROTOCOL SPECIFICATION

The Bluetooth logical link control and adaptation protocol (L2CAP) supports higher level protocol multiplexing, packet segmentation and reassembly, and the conveying of quality of service information. The protocol state machine, packet format and composition are described in this document.



CONTENTS

1 Introduction 21
1.1 L2CAP Features 21
1.2 Assumptions 25
1.3 Scope 25
1.4 Terminology 26
2 General Operation 29
2.1 Channel Identifiers 29
2.2 Operation Between Devices 29
2.3 Operation Between Layers 31
2.4 Modes of Operation 31
3 Data Packet Format 33
3.1 Connection-oriented Channel in Basic L2CAP Mode 33
3.2 Connectionless Data Channel in Basic L2CAP Mode 34
3.3 Connection-oriented Channel in Retransmission/Flow Control Modes 35
3.3.1 L2CAP header fields 35
3.3.2 Control field (2 octets) 36
3.3.3 L2CAP SDU length field (2 octets) 38
3.3.4 Information payload field (0 to 65531 octets) 38
3.3.5 Frame check sequence (2 octets) 39
3.3.6 Invalid frame detection 40
4 Signalling Packet Formats 41
4.1 Command Reject (code 0x01) 43
4.2 Connection Request (code 0x02) 44
4.3 Connection Response (code 0x03) 46
4.4 Configuration Request (code 0x04) 47
4.5 Configuration Response (code 0x05) 50
4.6 Disconnection Request (code 0x06) 52
4.7 Disconnection Response (code 0x07) 53
4.8 Echo Request (code 0x08) 53
4.9 Echo Response (code 0x09) 54
4.10 Information Request (code 0x0A) 54
4.11 Information Response (code 0x0B) 55
4.12 Extended Feature Mask 56
5 Configuration Parameter Options 57
5.1 Maximum Transmission Unit (MTU) 57
5.2 Flush Timeout Option 59
5.3 Quality of Service (QoS) Option 60

5.4 Retransmission and Flow Control Option 64

6 State Machine 67

6.1 General rules for the state machine: 67

6.1.1 CLOSED state 68

6.1.2 WAIT_CONNECT_RSP state 69

6.1.3 WAIT_CONNECT state 69

6.1.4 CONFIG state 70

6.1.5 OPEN state 73

6.1.6 WAIT_DISCONNECT state 73

6.2 Timers events 75

6.2.1 RTX 75

6.2.2 ERTX 76

7 General Procedures 79

7.1 Configuration Process 79

7.1.1 Request path 80

7.1.2 Response path 80

7.2 Fragmentation and Recombination 81

7.2.1 Fragmentation of L2CAP PDUs 81

7.2.2 Recombination of L2CAP PDUs 82

7.3 Encapsulation of SDUs 83

7.3.1 Segmentation of L2CAP SDUs 83

7.3.2 Reassembly of L2CAP SDUs 84

7.3.3 Segmentation and fragmentation 84

7.4 Delivery of Erroneous L2CAP SDUs 85

7.5 Operation with Flushing 85

7.6 Connectionless Data Channel 86

8 Procedures for Flow Control and Retransmission 87

8.1 Information Retrieval 87

8.2 Function of PDU Types for Flow Control and Retransmission 87

8.2.1 Information frame (I-frame) 87

8.2.2 Supervisory Frame (S-frame) 87

8.2.2.1 Receiver Ready (RR) 87

8.2.2.2 Reject (REJ) 88

8.3 Variables and Sequence Numbers 89

8.3.1 Sending peer 89

8.3.1.1 Send sequence number TxSeq 89

8.3.1.2 Send state variable NextTXSeq 89

8.3.1.3 Acknowledge state variable ExpectedAckSeq 90

8.3.2 Receiving peer 91

8.3.2.1 Receive sequence number ReqSeq 91

8.3.2.2 Receive state variable, ExpectedTxSeq 91
8.3.2.3 Buffer state variable BufferSeq 91
8.4 Retransmission Mode 93
8.4.1 Transmitting frames 93
8.4.1.1 Last received R was set to zero 93
8.4.1.2 Last received R was set to one 95
8.4.2 Receiving I-frames 95
8.4.3 I-frames pulled by the SDU reassembly function 95
8.4.4 Sending and receiving acknowledgements 96
8.4.4.1 Sending acknowledgements 96
8.4.4.2 Receiving acknowledgements 96
8.4.5 Receiving REJ frames 97
8.4.6 Waiting acknowledgements 97
8.4.7 Exception conditions 97
8.4.7.1 TxSeq Sequence error 97
8.4.7.2 ReqSeq Sequence error 98
8.4.7.3 Timer recovery error 98
8.4.7.4 Invalid frame 98
8.5 Flow Control Mode 99
8.5.1 Transmitting I-frames 99
8.5.2 Receiving I-frames 100
8.5.3 I-frames pulled by the SDU reassembly function 100
8.5.4 Sending and receiving acknowledgements 100
8.5.4.1 Sending acknowledgements 100
8.5.4.2 Receiving acknowledgements 100
8.5.5 Waiting acknowledgements 101
8.5.6 Exception conditions 101
8.5.6.1 TxSeq Sequence error 101
8.5.6.2 ReqSeq Sequence error 102
8.5.6.3 Timer recovery error 102
8.5.6.4 Invalid frame 102
9 List of Figures 103
10 List of Tables 105
11 Appendix 105

1 INTRODUCTION

This section of the Bluetooth Specification defines the Logical Link Control and Adaptation Layer Protocol, referred to as L2CAP. L2CAP is layered over the Link Controller Protocol and resides in the data link layer as shown in Figure 1.1. L2CAP provides connection-oriented and connectionless data services to upper layer protocols with protocol multiplexing capability, segmentation and reassembly operation, and group abstractions. L2CAP permits higher level protocols and applications to transmit and receive upper layer data packets (L2CAP Service Data Units, SDU) up to 64 kilobytes in length. L2CAP also permits per-channel flow control and retransmission via the Flow Control and Retransmission Modes.

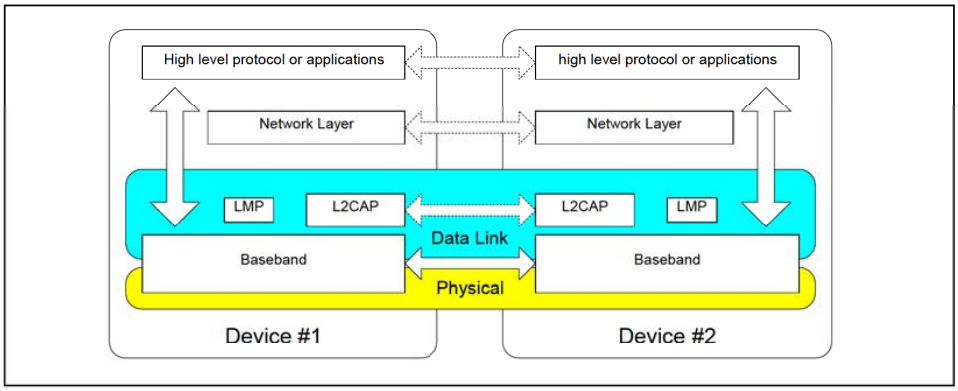


Figure 1.1: L2CAP within protocol layers

The L2CAP layer provides logical channels, named L2CAP channels, which are mapped to L2CAP logical links supported by an ACL logical transport, see baseband specification [vol.3, part B] Section 4.4 on page 98.

1.1 L2CAP FEATURES

The functional requirements for L2CAP include protocol/channel multiplexing, segmentation and reassembly (SAR), per-channel flow control, error control and group management. Figure 1.2 on page 22 illustrates how L2CAP data flows fit into the Bluetooth Protocol Stack. L2CAP lies above the Link Controller Protocol and interfaces with other communication protocols such as the Bluetooth Service Discovery Protocol (SDP), RFCOMM, Telephony Control (TCS) and Bluetooth Network Encapsulation Protocol (BNEP). Voice-quality channels for audio and telephony applications and synchronous transparent connections are usually run over synchronous logical transports, see [vol.3, part B] Section 4.3 on page 98. Packetized audio data, such as IP Telephony, may be sent using communication protocols running over L2CAP.

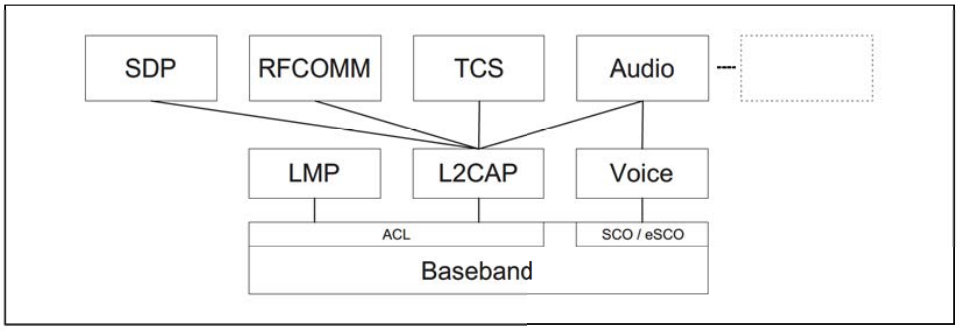


Figure 1.2: L2CAP data flows in Bluetooth Protocol Architecture

Figure 1.3 on page 22 breaks down L2CAP into its architectural components. The Channel Manager provides the control plane functionality and is responsible for all internal signalling, L2CAP peer-to-peer signalling and signalling with higher and lower layers. It performs the state machine functionality described in Section 6 on page 67 and uses message formats described in Section 4 on page 41, Section 5 on page 57. The Retransmission and Flow Control block provides per-channel flow control and optional retransmission for applications that require it. The Resource Manager is responsible for providing a frame relay service to the Channel Manager, the Retransmission and Flow Control block and those application data streams that do not require Retransmission and Flow Control services. It is responsible for coordinating the transmission and reception of packets related to multiple L2CAP channels over the facilities offered at the lower layer interface.

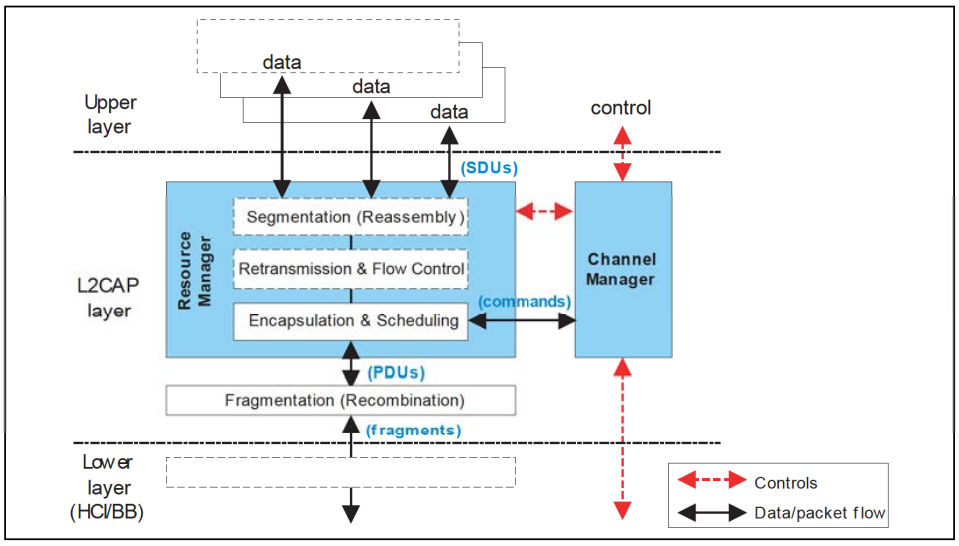


Figure 1.3: L2CAP architectural blocks

- *Protocol/channel multiplexing*

  L2CAP supports multiplexing because the Baseband Protocol does not support any 'type' field identifying the higher layer protocol being multiplexed above it.

  During channel setup, protocol multiplexing capability is used to route the connection request to the correct upper layer protocol.

  For data transfer, logical channel multiplexing is needed to distinguish between multiple upper layer entities. There may be more than one upper layer entity using the same protocol.

- *Segmentation and reassembly*

  With the frame relay service offered by the Resource Manager, the length of transport frames is controlled by the individual applications running over L2CAP. Many multiplexed applications are better served if L2CAP has control over the PDU length. This provides the following benefits:

  a) Segmentation will allow the interleaving of application data units in order to satisfy latency requirements.

  b) Memory and buffer management is easier when L2CAP controls the packet size.

  c) Error correction by retransmission can be made more efficient.

  d) The amount of data that is destroyed when an L2CAP PDU is corrupted or lost can be made smaller than the application's data unit.

  e) The application is decoupled from the segmentation required to map the application packets into the lower layer packets.

- *Flow control per L2CAP channel*

  When several data streams run over the same L2CAP logical link, using separate L2CAP channels, each channel may require individual flow control. Also L2CAP provides flow control services to profiles or applications that need flow control and can avoid having to implement it. Due to the delays between the L2CAP layers, stop-and-go flow control as employed in the baseband is not sufficient. A window based flow control scheme is provided. The use of flow control is an optional aspect of the L2CAP protocol.

- *Error control and retransmissions*

  Some applications require a residual error rate much smaller than the baseband can deliver. L2CAP includes optional error checks and retransmissions of L2CAP PDUs. The error checking in L2CAP protects against errors due to the baseband falsely accepting packet headers and due to failures of the HEC or CRC error checks on the baseband packets. Retransmission Mode also protects against loss of packets due to flush on the same logical transport. The error control works in conjunction with flow control in the sense that the flow control mechanism will throttle retransmissions as well as first transmissions. The use of error control and retransmission procedures is optional.

- *Fragmentation and Recombination*

  The lower layers have limited transmission capabilities and may require fragment sizes different from those created by L2CAP segmentation. Therefore layers below L2CAP may further fragment and recombine L2CAP PDUs to create fragments which fit each layer capabilities. During transmission of an L2CAP PDU, many different levels of fragmentation and recombination may occur in both peer devices.

  The HCI driver or controller may fragment L2CAP PDUs to honor packet size constraints of a host controller interface transport scheme. This results in HCI data packet payloads carrying start and continuation fragments of the L2CAP PDU. Similarly the link controller may fragment L2CAP PDUs to map them into baseband packets. This results in baseband packet payloads carrying start and continuation fragments of the L2CAP PDU.

  Each layer of the protocol stack may pass on different sized fragments of L2CAP PDUs, and the size of fragments created by a layer may be different in each peer device. However the PDU is fragmented within the stack, the receiving L2CAP entity still recombines the fragments to obtain the original L2CAP PDU.

- *Quality of Service*

  The L2CAP connection establishment process allows the exchange of information regarding the quality of service (QoS) expected between two Bluetooth devices. Each L2CAP implementation monitors the resources used by the protocol and ensures that QoS contracts are honored.

1.2 ASSUMPTIONS

The protocol is designed based on the following assumptions:

1. The ACL logical transport and L2CAP logical link between two devices is set up using the Link Manager Protocol. The baseband provides orderly delivery of data packets, although there might be individual packet corruption and duplicates. No more than 1 unicast ACL logical transport exists between any two devices.
2. The baseband always provides the impression of full-duplex communication channels. This does not imply that all L2CAP communications are bi-directional. Multicasts and unidirectional traffic (e.g., video) do not require duplex channels.
3. The L2CAP layer provides a channel with a degree of reliability based on the mechanisms available at the baseband layer and with optional additional packet segmentation and error detection that can be enabled in the enhanced L2CAP layer. The baseband performs data integrity checks and resends data until it has been successfully acknowledged or a timeout occurs. Because acknowledgements may be lost, timeouts may occur even after the data has been successfully sent. The link controller protocol uses a 1-bit sequence number. Note that the use of baseband broadcast packets is prohibited if reliability is required, since all broadcasts start the first segment of an L2CAP packet with the same sequence bit.
4. Some applications will expect independent flow control, independence from the effects of other traffic and, in some cases, better error control than the baseband provides. The Flow and Error Control block provides two modes. Retransmission Mode offers segmentation, flow control and L2CAP PDU retransmissions. Flow control mode offers just the segmentation and flow control functions. If Basic L2CAP mode is chosen, the Flow and Error Control block is not used.

1.3 SCOPE

The following features are outside the scope of L2CAP's responsibilities:

- L2CAP does not transport audio or transparent synchronous data designated for SCO or eSCO logical transports.
- L2CAP does not support a reliable multicast channel. See Section 3.2 on page 34.
- L2CAP does not support the concept of a global group name.

1.4 TERMINOLOGY

The following formal definitions apply:

Term	Description
Upper layer	The system layer above the L2CAP layer, which exchanges data with L2CAP in the form of SDUs. The upper layer may be represented by an application or higher protocol entity known as the Service Level Protocol. The interface of the L2CAP layer with the upper layer is not specified.
Lower layer	The system layer below the L2CAP layer, which exchanges data with the L2CAP layer in the form of PDUs, or fragments of PDUs. The lower layer is mainly represented within the Bluetooth Controller, however a Host Controller Interface (HCI) may be involved, such that an HCI host driver could also be seen as the lower layer. Except for the HCI functional specification (in case HCI is involved) the interface between L2CAP and the lower layer is not specified.
L2CAP channel	The logical connection between two endpoints in peer devices, characterized by their Channel Identifiers (CID), which is multiplexed on the L2CAP logical link, which is supported by an ACL logical transport, see [vol.3, part B] Section 4.4 on page 98
SDU, or L2CAP SDU	Service Data Unit: a packet of data that L2CAP exchanges with the upper layer and transports transparently over an L2CAP channel using the procedures specified here. The term SDU is associated with data originating from upper layer entities only, i.e. does not include any protocol information generated by L2CAP procedures.
Segment, or SDU segment	A part of an SDU, as resulting from the Segmentation procedure. An SDU may be split into one or more segments. Note: this term is relevant only to the Retransmission Mode and Flow Control Mode, not to the Basic L2CAP Mode.
Segmentation	A procedure used in the L2CAP Retransmission and Flow Control Modes, resulting in an SDU being split into one or more smaller units, called Segments, as appropriate for the transport over an L2CAP channel. Note: this term is relevant only to the Retransmission Mode and Flow Control Mode, not to the Basic L2CAP Mode.
Reassembly	The reverse procedure corresponding to Segmentation, resulting in an SDU being re-established from the segments received over an L2CAP channel, for use by the upper layer. Note that the interface between the L2CAP and the upper layer is not specified; therefore, reassembly may actually occur within an upper layer entity although it is conceptually part of the L2CAP layer. Note: this term is relevant only to the Retransmission Mode and Flow Control Mode, not to the Basic L2CAP Mode.
PDU, or L2CAP PDU	Protocol Data Unit a packet of data containing L2CAP protocol information fields, control information, and/or upper layer information data. A PDU is always started by a Basic L2CAP header. Types of PDUs are: B-frames, I-frames, S-frames, C-frames and G-frames.

Table 1.1: Terminology

Term	Description
Basic L2CAP header	Minimum L2CAP protocol information that is present in the beginning of each PDU: a length field and a field containing the Channel Identifier (CID)
Basic information frame (B-frame)	A B-frame is a PDU used in the Basic L2CAP mode for L2CAP data packets. It contains a complete SDU as its payload, encapsulated by a basic L2CAP header.
Information frame (I-frame)	An I-frame is a PDU used in the Retransmission Mode and the Flow Control Mode. It contains an SDU segment and additional protocol information, encapsulated by a basic L2CAP header
Supervisory frame (S-frame)	An S-frame is a PDU used in the Retransmission Mode and the Flow Control Mode. It contains protocol information only, encapsulated by a basic L2CAP header, and no SDU data.
Control frame (C-frame)	A C-frame is a PDU that contains L2CAP signalling messages exchanged between the peer L2CAP entities. C-frames are exclusively used on the L2CAP signalling channel.
Group frame (G-frame)	G-frame is a PDU exclusively used on Connectionless L2CAP channels in the Basic L2CAP mode. It contains a complete SDU as its payload, encapsulated by a specific header.
Fragment	A part of a PDU, as resulting from a fragmentation operation. Fragments are used only in the delivery of data to and from the lower layer. They are not used for peer-to-peer transportation. A fragment may be a Start or Continuation Fragment with respect to the L2CAP PDU. A fragment does not contain any protocol information beyond the PDU; the distinction of start and continuation fragments is transported by lower layer protocol provisions. Note: Start Fragments always begin with the Basic L2CAP header of a PDU.
Fragmentation	A procedure used to split L2CAP PDUs to smaller parts, named fragments, appropriate for delivery to the lower layer transport. Although described within the L2CAP layer, fragmentation may actually occur in an HCI host driver, and/or in the Controller, to accommodate the L2CAP PDU transport to HCI data packet or baseband packet sizes. Fragmentation of PDUs may be applied in all L2CAP modes. Note: in version 1.1, Fragmentation and Recombination was referred to as “Segmentation and Reassembly“.
Recombination	The reverse procedure corresponding to fragmentation, resulting in an L2CAP PDU re-established from fragments. In the receive path, full or partial recombination operations may occur in the Controller and/or the Host, and the location of recombination does not necessarily correspond to where fragmentations occurs on the transmit side.
Maximum Transmission Unit (MTU)	The maximum size of payload data, in octets, that the upper layer entity is capable of accepting, i.e. the MTU corresponds to the maximum SDU size.

Table 1.1: Terminology

Term	Description
Maximum PDU payload Size (MPS)	The maximum size of payload data in octets that the L2CAP layer entity is capable of accepting, i.e. the MPS corresponds to the maximum PDU payload size. Note: in the absence of segmentation, or in the Basic L2CAP Mode, the Maximum Transmission Unit is the equivalent to the Maximum PDU payload Size and shall be made equal in the configuration parameters.
Signalling MTU (MTU_{sig})	The maximum size of command information that the L2CAP layer entity is capable of accepting. The MTU_{sig}, refers to the signalling channel only and corresponds to the maximum size of a C-frame, excluding the Basic L2CAP header. The MTU_{sig} value of a peer is discovered when a C-frame that is too large is rejected by the peer.
Connectionless MTU (MTU_{cnl})	The maximum size of the connection packet information that the L2CAP layer entity is capable of accepting. The MTU_{cnl} refers to the connectionless channel only and corresponds to the maximum G-frame, excluding the Basic L2CAP header. The MTU_{cnl} of a peer can be discovered by sending an Information Request.
MaxTransmit	In Retransmission mode, MaxTransmit controls the number of transmissions of a PDU that L2CAP is allowed to try before assuming that the PDU (and the link) is lost. The minimum value is 1 (only 1 transmission permitted). Note: Setting MaxTransmit to 1 prohibits PDU retransmissions. Failure of a single PDU will cause the link to drop. By comparison, in Flow Control mode, failure of a single PDU will not necessarily cause the link to drop.

Table 1.1: Terminology

2 GENERAL OPERATION

L2CAP is based around the concept of *'channels'*. Each one of the endpoints of an L2CAP channel is referred to by a *channel identifier (CID)*.

2.1 CHANNEL IDENTIFIERS

A channel identifier (CID) is the local name representing a logical channel endpoint on the device. The null identifier (0x0000) is an illegal identifier and shall never be used as a destination endpoint. Identifiers from 0x0001 to 0x003F are reserved for specific L2CAP functions. Implementations are free to manage the remaining CIDs in a manner best suited for that particular implementation, with the provision that two simultaneously active L2CAP channels shall not share the same CID. Table 2.1 on page 29 summarizes the definition and partitioning of the CID name space.

CID assignment is relative to a particular device and a device can assign CIDs independently from other devices (unless it needs to use any of the reserved CIDs shown in the table below). Thus, even if the same CID value has been assigned to (remote) channel endpoints by several remote devices connected to a single local device, the local device can still uniquely associate each remote CID with a different device.

CID	Description
0x0000	Null identifier
0x0001	Signalling channel
0x0002	Connectionless reception channel
0x0003-0x003F	Reserved
0x0040-0xFFFF	Dynamically allocated

Table 2.1: CID name space

2.2 OPERATION BETWEEN DEVICES

Figure 2.1 on page 30 illustrates the use of CIDs in a communication between corresponding peer L2CAP entities in separate devices. The connection-oriented data channels represent a connection between two devices, where a CID identifies each endpoint of the channel. The connectionless channels restrict data flow to a single direction. These channels are used to support a channel 'group' where the CID on the source represents one or more remote devices. There are also a number of CIDs reserved for special purposes. The signalling channel is one example of a reserved channel. This channel is used to create and establish connection-oriented data channels and to negotiate changes in the characteristics of connection oriented and connectionless channels. Support for a signalling channel within an L2CAP entity is mandatory.

Note: it is assumed that an L2CAP signalling channel is available immediately when an ACL logical transport is established between two devices, and L2CAP traffic is enabled on the L2CAP logical link.
Another CID is reserved for all incoming connectionless data traffic. In the example below, a CID is used to represent a group consisting of device #3 and #4. Traffic sent from this channel ID is directed to the remote channel reserved for connectionless data traffic.

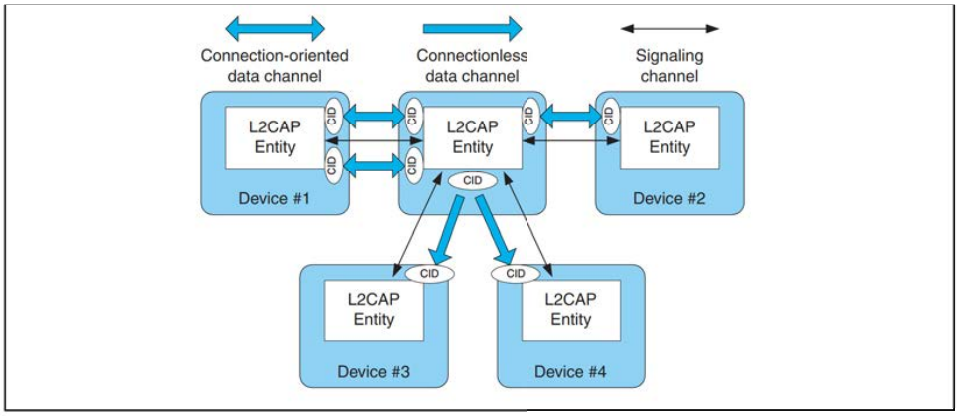


Figure 2.1: Channels between devices

Table 2.2 on page 30 describes the various channels and their source and destination identifiers. A CID is allocated to identify the local endpoint and shall be in the range 0x0040 to 0xFFFF. Section 6 on page 67 describes the state machine associated with each connection-oriented channel. Section 3.1 on page 33 describes the packet format associated with bi-directional channels and Section 3.2 on page 34 describes the packet format associated with uni-directional channels.

Channel Type	Local CID (sending)	Remote CID (receiving)
Connection-oriented	Dynamically allocated	Dynamically allocated
Connectionless data	Dynamically allocated	0x0002 (fixed)
Signalling	0x0001 (fixed)	0x0001 (fixed)

Table 2.2: Types of Channel Identifiers

2.3 OPERATION BETWEEN LAYERS

L2CAP implementations should follow the general architecture described below. L2CAP implementations transfer data between upper layer protocols and the lower layer protocol. This document lists a number of services that should be exported by any L2CAP implementation. Each implementation shall also support a set of signalling commands for use between L2CAP implementations. L2CAP implementations should also be prepared to accept certain types of events from lower layers and generate events to upper layers. How these events are passed between layers is implementation specific.

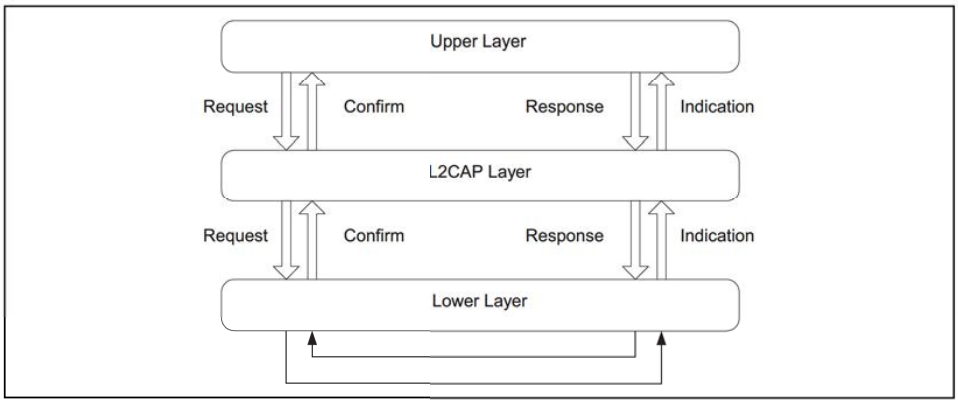


Figure 2.2: L2CAP transaction model.

2.4 MODES OF OPERATION

L2CAP may operate in one of three different modes as selected for each L2CAP channel by an upper layer.

The modes are:

- Basic L2CAP Mode (equivalent to L2CAP specification in Bluetooth v1.1) [1]
- Flow Control Mode
- Retransmission Mode

The modes are enabled using the configuration procedure. The Basic L2CAP Mode is the default mode, which is used when no other mode is agreed.

In Flow Control and Retransmission modes, PDUs exchanged with a peer entity are numbered and acknowledged. The sequence numbers in the PDUs are used to control buffering, and a TxWindow size is used to limit the required buffer space and/or provide a method for flow control. In addition to the window size, the Token Bucket size parameter of the flow specification can be used to

1. Specification of the Bluetooth System v1.1 (Feb 22nd 2001): volume 1, part D.

dimension the buffers; in particular on channels that do not use flow and error control.

In Flow Control Mode no retransmissions take place, but missing PDUs are detected and can be reported as lost.

In Retransmission Mode a timer is used to ensure that all PDUs are delivered to the peer, by retransmitting PDUs as needed. A go-back-n repeat mechanism is used to simplify the protocol and limit the buffering requirements.

3 DATA PACKET FORMAT

L2CAP is packet-based but follows a communication model based on *channels*. A channel represents a data flow between L2CAP entities in remote devices. Channels may be connection-oriented or connectionless.
All packet fields shall use Little Endian byte order.

3.1 CONNECTION-ORIENTED CHANNEL IN BASIC L2CAP MODE

Figure 3.1 on page 33 illustrates the format of the L2CAP PDU within a connection-oriented channel. In basic L2CAP mode, the L2CAP PDU on a connection-oriented channel is also referred to as a "B-frame".

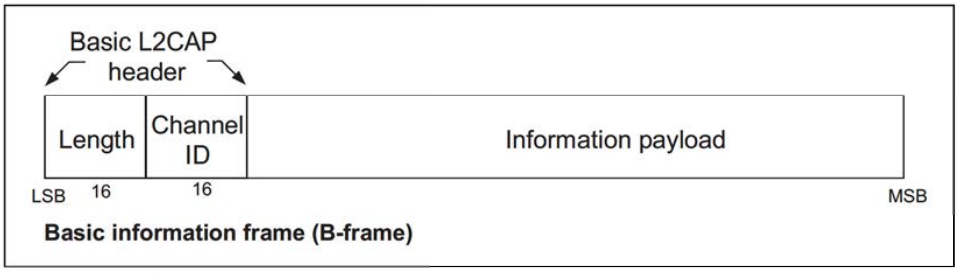


Figure 3.1: PDU format in Basic L2CAP mode on connection-oriented channels (field sizes in bits)

The fields shown are:

- *Length: 2 octets (16 bits)*

  Length indicates the size of the information payload in octets, excluding the length of the L2CAP header. The length of an information payload can be up to 65535 octets. The Length field is used for recombination and serves as a simple integrity check of the recombined L2CAP packet on the receiving end.

- *Channel ID: 2 octets*

  The channel ID (CID) identifies the destination channel endpoint of the packet.

- *Information payload: 0 to 65535 octets*

  This contains the payload received from the upper layer protocol (outgoing packet), or delivered to the upper layer protocol (incoming packet). The MTU is determined during channel configuration (see Section 5.1 on page 57). The minimum supported MTU for the signalling PDUs (MTU_{sig}) is 48 octets (see Section 4 on page 41).

3.2 CONNECTIONLESS DATA CHANNEL IN BASIC L2CAP MODE

Figure 3.2 illustrates the L2CAP PDU format within a connectionless data channel. Here, the L2CAP PDU is also referred to as a "G-frame".

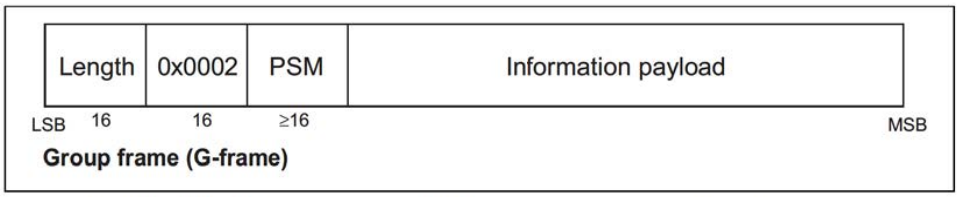


Figure 3.2: L2CAP PDU format in Basic L2CAP mode on Connectionless channel

The fields shown are:

- *Length: 2 octets*

  Length indicates the size of information payload plus the PSM field in octets.
- *Channel ID: 2 octets*

  Channel ID (0x0002) reserved for connectionless traffic.
- *Protocol/Service Multiplexer (PSM): 2 octets (minimum)*

  For information on the PSM field see Section 4.2 on page 44.
- *Information payload: 0 to 65533 octets*

  The payload information to be distributed to all members of the piconet. Implementations shall support a connectionless MTU (MTU_{cnl}) of 48 octets on connectionless channels. Devices may also explicitly change to a larger or smaller connectionless MTU (MTU_{cnl}).

  Note: the maximum size of the Information payload field decreases accordingly if the PSM field is extended beyond the two octet minimum.

3.3 CONNECTION-ORIENTED CHANNEL IN RETRANSMISSION/FLOW CONTROL MODES

To support flow control and retransmissions, L2CAP PDU types with protocol elements in addition to the Basic L2CAP header are defined. The information frames (I-frames) are used for information transfer between L2CAP entities. The supervisory frames (S-frames) are used to acknowledge I-frames and request retransmission of I-frames.

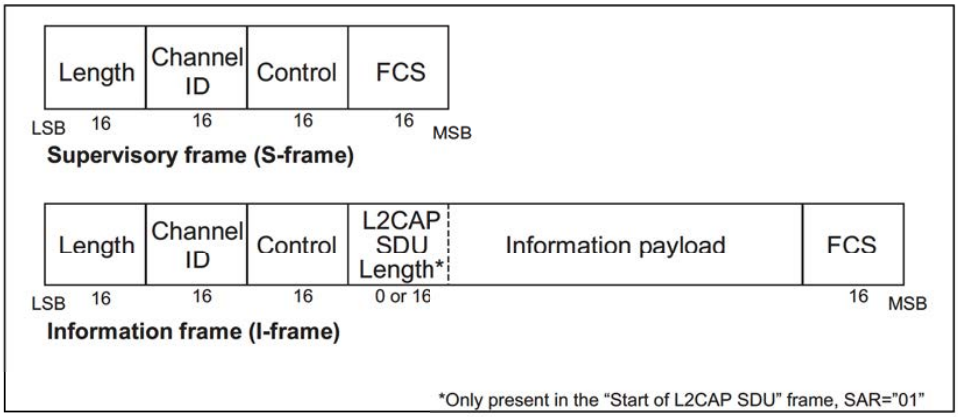


Figure 3.3: L2CAP PDU formats in Flow Control and Retransmission Modes

3.3.1 L2CAP header fields

- *Length: 2 octets*

  The first two octets in the L2CAP PDU contain the length of the entire L2CAP PDU in octets, excluding the Length and CID field.

  For I-frames and S-frames, the Length field therefore includes the octet lengths of the Control, L2CAP SDU Length (when present), Information octets and frame check sequence (FCS) fields.

  If the L2CAP SDU length field is present then the maximum number of Information octets in one I-frame is 65529 octets. If the L2CAP SDU length field is not present then the maximum number of Information octets in one I-frame is 65531 octets.

- *Channel ID: 2 octets*

  This field contains the Channel Identification (CID).