

J-Link>setbp 0x1140 <--- function exit
Breakpoint set @ addr 0x00001140 (Handle = 1)

```
PC = 00001140, CycleCnt = 02ACD568
R0 = 00000031, R1 = 00000002, R2 = 00000010, R3 = 38000000
R4 = 00000000, R5 = 00000000, R6 = 00000000, R7 = 2003FFE8
R8 = 9220C050, R9 = D08C106E, R10= 20030000, R11= 00000000
R12= ABA08801
SP(R13)= 2003FFE8, MSP= 2003FFE8, PSP= 00000000, R14(LR) = 0000113B
J-Link>mem32 2003FFE8,0x20
2003FFE8 = 00000066 2004000B 24070001 24242424
2003FFF8 = BF00BF00 BF00BF00 BF00BF00 BF00BF00
20040008 = E7FEBF00 A0F10400 33338047 33333131
20040018 = 33333131 33333131 33333131 33333131
20040028 = 33333131 33333131 33333131 33333131
20040038 = 33333131 33333131 33333131 33333131
20040048 = 33333131 33333131 33333131 33333131
20040058 = 33333131 33333131 33333131 33333131
J-Link>s
00001140: 80 BD          POP      {R7,PC}
```

J-Link>regs
PC = 2004000A, CycleCnt = 02ACD570
R0 = 00000031, R1 = 00000002, R2 = 00000010, R3 = 38000000
R4 = 00000000, R5 = 00000000, R6 = 00000000, R7 = 00000066
R8 = 9220C050, R9 = D08C106E, R10= 20030000, R11= 00000000
R12= ABA08801
SP(R13)= 2003FFF0, MSP= 2003FFF0, PSP= 00000000, R14(LR) =
J-Link>s
2004000A: FE E7 B #-0x04
J-Link>s
2004000A: FE E7 B #-0x04





```
100     QSPI_Read_Product_Headers((struct_configuration_script_ptr *)&configuration_script, product_img_offsets);
101 }
102 while ( BYTE1(configuration_script) != 1 );
103 WDOG_feed_ff();
104 if ( check_current_fw_addr_and_update_addr(product_img_offsets) )// differing offsets mean we have an update, enter
105 {
106     if ( !SECUREBOOT_check_and_validate(
107         product_img_offsets[1],
108         (FW_ImageHeader *)&dword_2003C950,
109         (int)product_img_offsets,
110         (struct_configuration_script_ptr *)&configuration_script) )// / returns 1 if secure boot is not enabled
111     {
112         QSPI_process_product_update((struct_configuration_script_ptr *)&configuration_script, product_img_offsets); // re
113         WDOG_Pet_1c34();
114     }
115 }
116
117 }
118 }
119
120
121
122 else if ( !SECUREBOOT_check_and_validate(
123     product_img_offsets[0],
124     (FW_ImageHeader *)&dword_2003C950,
125     (int)product_img_offsets,
126     (struct_configuration_script_ptr *)&configuration_script) )// returns 1 if secure boot is not enabled
127 {
128     WDOG_Pet_1c34(); // we hit this
129 }
130 SECUREBOOT_CHECK_key_revocation(dword_2003C950);
131 sub_A6E((struct_configuration_script_ptr *)&configuration_script, product_img_offsets);
132 WDOG_feed_ff();
133 if ( !SECUREBOOT_CHECK_setup_QSPIC_CTR_195C(
134     (struct_configuration_script_ptr *)&configuration_script,
135     product_img_offsets) )
136     WDOG_Pet_1c34();
137 OTPC_standby();
138 Cache_setup_qspi_cache(dword_2003C950, product_img_offsets);
139 write_to_sysram(dword_2003C950, product_img_offsets);
140 return RESET_to_REMAP_ADR_val(2u); // SW Reset to QSPI Flash
141 }
```