

# Lab 1: Preparation

## Question 2

0x0000067E	4806	LDR r0, [pc, #24] ; @0x0000068C	<pre>while(1){     PF1 ^= 0x02; }</pre>
0x00000680	6880	LDR r0, [r0, #0x08]	
0x00000682	F0800002	EOR r0, r0, #0x02	
0x00000686	4904	LDR r1, [pc, #16] ; @0x0000068C	
0x00000688	6088	STR r0, [r1, #0x08]	
0x0000068A	E7F8	B 0x0000067E	
0x0000068C	E608	DCW 0xE608	
0x0000068E	400F	DCW 0x400F	
0x00000684	5400	DCW 0x5400	
0x00000686	4002	DCW 0x4002	
0x00000688	551C	DCW 0x551C	
0x0000068A	4002	DCW 0x4002	
0x0000068C	5000	DCW 0x5000	
0x0000068E	4002	DCW 0x4002	

### What is the purpose of all the DCW statements?

The purpose of the DCW statements is to allocate a 4 bytes of data into that memory location. This is an assembler directive that stores the data before the program is run. Padding is added to ensure memory alignment. In this example, the DCW statements are used to store the address of PF1 in memory so that the program can access it with offsets.

### How fast will PF1 toggle?

It takes 25ns to execute an instruction (on average), and the main loop has 6 instructions, so it takes  $25 \times 6 = 150\text{ns}$  to toggle.

### What is in R0 after the first LDR? What is in R0 after the second LDR?

R0 will have 0x4002500 after the first LDR, which is the address of PF1 for bit-specific addressing. R0 will have the current value of PF1 after the second LDR.

### How would you have written the compiler to remove an instruction?

Two of the LDR instructions are loading the same thing (0x4002500 – the address of PF1). The second one can be removed if the first one (in R0) was not overwritten by the value of PF1. The compiler could have used a different register to store the value of PF1 so that R0 can be used directly for the STR instruction.

### Is there a critical section?

There is not a critical section between the main function and the ISR because they are using bit-specific addressing. The ISR only read-modify-writes to PF2, while main does the same to PF1.