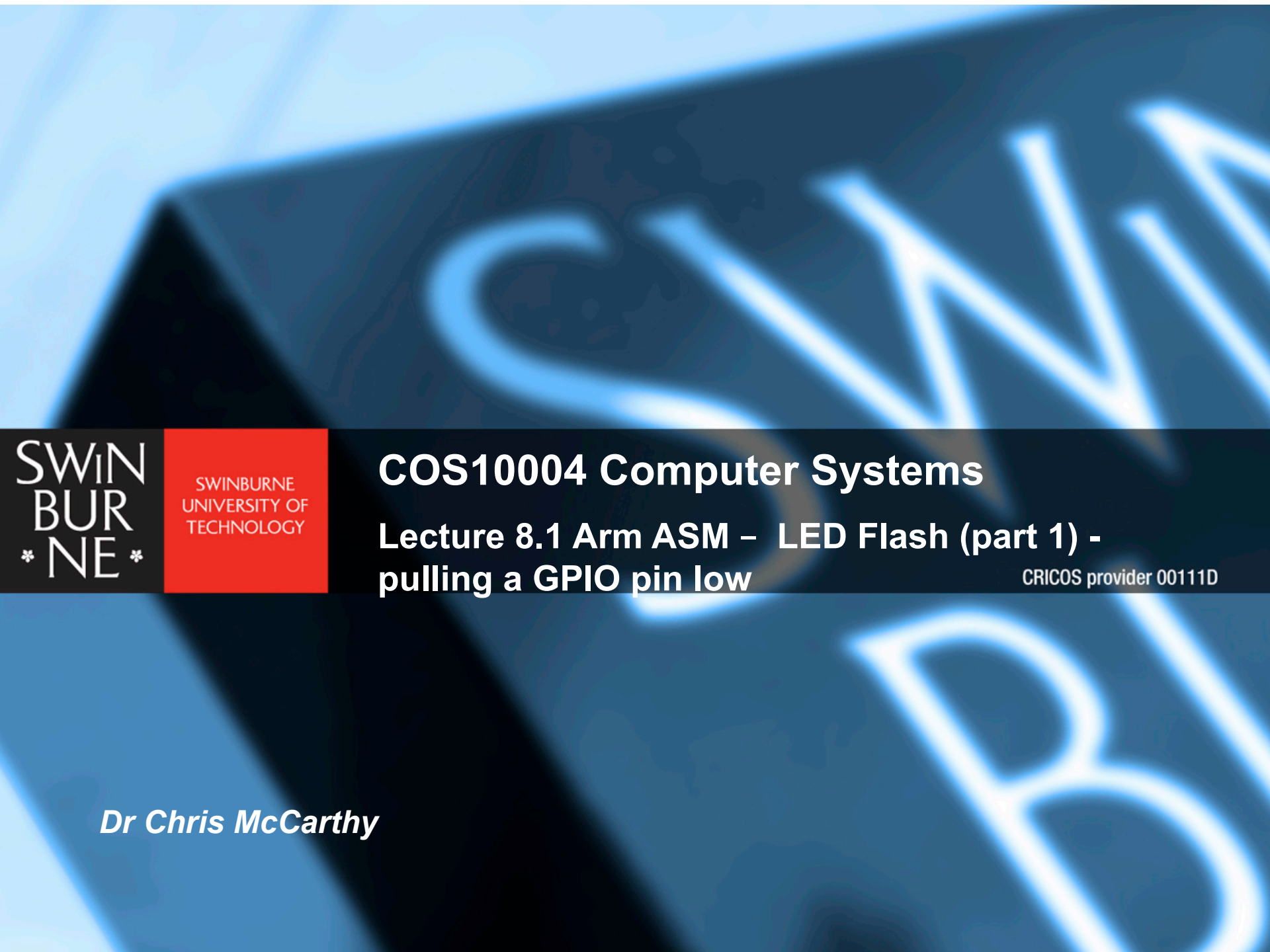


COS10004 Computer Systems

Lecture 8.1 Arm ASM – LED Flash (part 1) - pulling a GPIO pin low

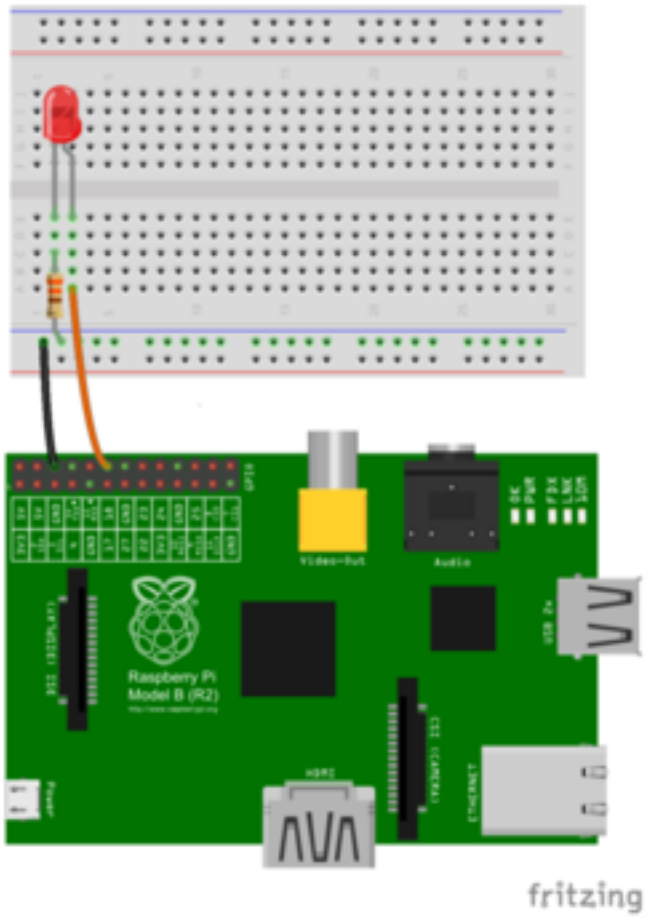
CRICOS provider 00111D

Dr Chris McCarthy

Our ARM ASM Agenda

- *assignment (mov, ldr)* 
- *arithmetic (add, sub, mul, ~~div~~)*
- *labels, branch (b)* 
- *registers, GPIO* 
- *selection (cmp, tst)* 
- *functions, parameters (bl)*
- *stack (push, pop)*
- *building up immediate constants*
- *ARM timer*
- *Turn on|off GPIO (gpio.s) (OK01, OK02)*

From Week 7



See https://www.youtube.com/watch?v=Rd9kvVs1lSQ for my tutorial on wiring this circuit

TURNING ON AN LED

```
BASE = $FE000000   ;  $ means HEX
GPIO_OFFSET=$200000

mov r0,BASE
orr r0,GPIO_OFFSET        ;r0 now equals 0xFE200000

mov r1,#1
lsl r1,#24                ;write 1 into r1, lsl 24 times to move the 1 to bit 24
str r1,[r0,#4]            ;write it into 5th (16/4+1)block of function register

mov r1,#1
lsl r1,#18                 ;write 1 into r1, lsl 18 times to move the 1 to bit 18
str r1,[r0,#28]           ;write it into first block of pull-up register

loop$:
b loop$                   ;loop forever
```

TURNING ON AN LED

```
BASE = $FE000000   ;  $ means HEX
GPIO_OFFSET=$200000

mov r0,BASE
orr r0,GPIO_OFFSET        ;r0 now equals 0xFE200000

mov r1,#1
lsl r1,#24                ;write 1 into r1, lsl 24 times to move the 1 to bit 24
str r1,[r0,#4]            ;write it into 5th (16/4+1)block of function register

mov r1,#1
lsl r1,#18                 ;write 1 into r1, lsl 18 times to move the 1 to bit 18
str r1,[r0,#28]           ;write it into first block of pull-up register

loop$:
b loop$                   ;loop forever
```

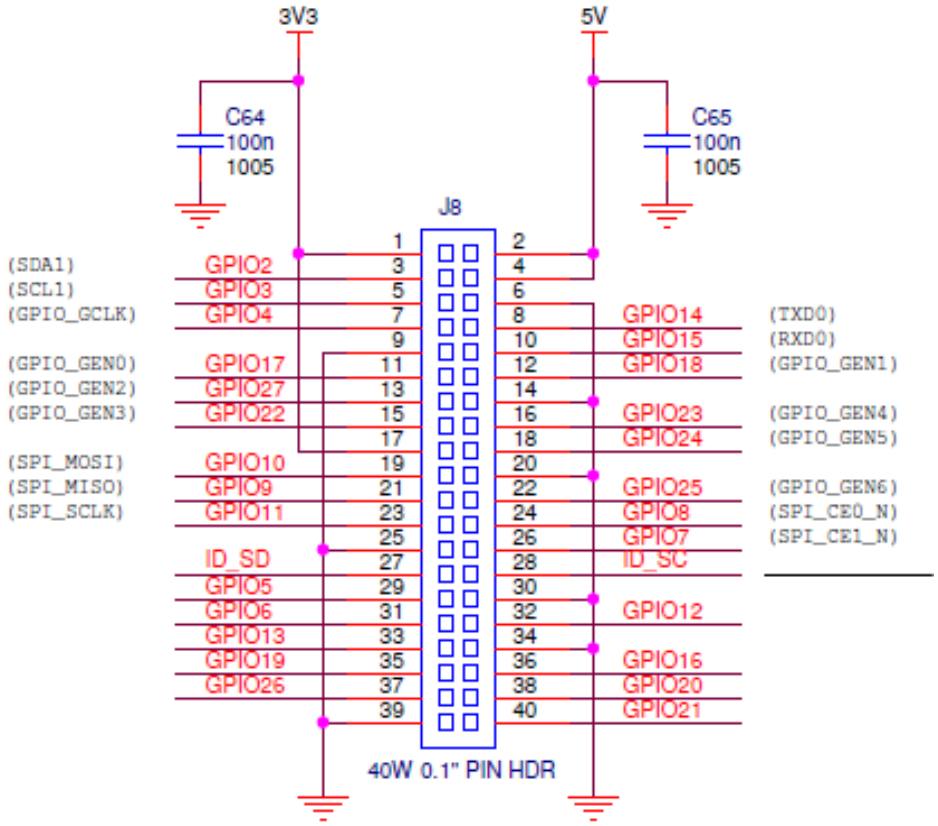
What about these numbers ?

Where did they come from
And what do they mean ?

These numbers all refer to settings and programming of the GPIO registers.

To understand this part of the Code we need to understand what the GPIO chip is, and how we interface with it to read to and write from the GPIO header pins.

GPIO LAYOUT FOR RPI 2B, 3B AND 4



GPIO EXPANSION

MAKING THE LED FLASH (OK2)

- pseudocode:

  Program the GPIO 18 for output

    Enable

      wait

      Disable

      wait

      repeat

We need to wait because the LED will flash too fast to see. The clock speed of the ARM – 700MHz – 1.2GHz.

Lab 8 Task resource on Canvas

MAKING THE LED FLASH (OK2)

- pseudocode:

  Program the GPIO 18 for output

    Enable

      wait

      Disable

      wait

      repeat

We need to wait because the LED will flash too fast to see. The clock speed of the ARM – 700MHz – 1.2GHz.

Lab 8 Task resource on Canvas

FUNCTIONALITY NEEDED

- Pull GPIO18 low (disable led)
- Insert delay between led enable and disable
- Lets deal with turning off LED first!

RECALL PSUEDO CODE

- store location of GPIO (BASE + GPIOADDR) in r0
- Enable “writing” for GPIO18 (our LED)
  - we need to set certain bits in the “function” register to program GPIO18 for writing
- Set output of GPIO18
  - Light on: set bit in the appropriate “write 1” register
  - Light off: set bit in the appropriate “write 0” register
- loop forever

RECALL PSUEDO CODE

- store location of GPIO (BASE + GPIOADDR) in r0
- Enable "writing" for GPIO18 (our LED)
  - we need to set certain bits in the "function" register to program GPIO18 for writing
- Set output of GPIO18
  - Light on: set bit in the appropriate "write 1" register
  - Light off: set bit in the appropriate "write 0" register
- loop forever

start of the BASE+ GPIO (RPi 2/3). Add this address to everything in the GPIO

1. SELECT FUNCTION

Each pin programmed by a 3-bit number. Those numbers are packed into 30 bits of each word

Hex	Offset (dec)	32-bit regsiters	Function Select Register			
0x3F200000	0	GPIO 0 - 9	**store GPIO start address:**	bits 0-2 = GPIO 0	bits 3-5 = GPIO 1	bits 6-8 = GPIO 2
	1		ldr r0,=0x3F200000	bits 9-11 = GPIO 3	bits 12-14 = GPIO 4	
	2			bits 15-17 = GPIO 5	bits 18-20 = GPIO 6	bits 21-23 = GPIO 7
	3			bits 24-26 = GPIO 8	bits 27-29 = GPIO 9	
0x3F200004	4	GPIO 10 - 19	**Enable Write: to GPIO18 (Lab 7)**	bits 0-2 = GPIO 10	bits 3-5 = GPIO 11	bits 6-8 = GPIO 12
	5		mov r1,#1	bits 9-11 = GPIO 13	bits 12-14 = GPIO 14	
	6		lsl r1,#24	bits 15-17 = GPIO 15	bits 18-20 = GPIO 16	bits 21-23 = GPIO 17
	7		str r1,[r0,#4]	bits 24-26 = GPIO 18	bits 27-29 = GPIO 19	
0x3F200008	8	GPIO 20 - 29	**Enable Read: from GPIO24(pin)**	bits 0-2 = GPIO 20	bits 3-5 = GPIO 21	bits 6-8 = GPIO 22
	9		mov r1,#0	bits 9-11 = GPIO 23	bits 12-14 = GPIO 24	
	10		lsl r1,#12	bits 15-17 = GPIO 25	bits 18-20 = GPIO 26	bits 21-23 = GPIO 27
	11		str r1,[r0,#8]	bits 24-26 = GPIO 28	bits 27-29 = GPIO 29	
0x3F20000C	12	GPIO 30 - 39		bits 0-2 = GPIO 30	bits 3-5 = GPIO 31	bits 6-8 = GPIO 32
	13			bits 9-11 = GPIO 33	bits 12-14 = GPIO 34	
	14			bits 15-17 = GPIO 35	bits 18-20 = GPIO 36	bits 21-23 = GPIO 37
	15			bits 24-26 = GPIO 38	bits 27-29 = GPIO 39	
0x3F200010	16	GPIO 40 - 49		bits 0-2 = GPIO 40	bits 3-5 = GPIO 41	bits 6-8 = GPIO 42
	17			bits 9-11 = GPIO 43	bits 12-14 = GPIO 44	
	18			bits 15-17 = GPIO 45	bits 18-20 = GPIO 46	bits 21-23 = GPIO 47
	19			bits 24-26 = GPIO 8	bits 27-29 = GPIO 49	
0x3F200014	20	GPIO 50 - 54	0-7 bits	bits 0-2 = GPIO 50	bits 3-5 = GPIO 51	bits 6-8 = GPIO 52
	21		8-15 bits	bits 9-11 = GPIO 53	bits 12-14 = GPIO 54	
	22		16-22 bits			
	23		23-29 bits			
0x3F200018	24					

Add 4 bytes (1 word) each time we go above 30 bits (10 GPIO pins)

2. Set value (R/W)

3 registers control writing 0, writing 1 or reading each pin.

Each pin programmed by a 1-bit number. 32 numbers are packed into 32 bits of each word.

This register writes 1 to the GPIO pin

0x3F20001C	28		**Write 1 to GPIO18 (Lab 7)**
	29	set bit n to turn ON GPIO n	mov r1,#1
	30		lsl r1,#18
	31		str r1,[r0,#28]
0x3F200020	**32**		
	33	set bit n to turn ON GPIO 32+n	
	34		
	35		
0x3F200024	36		

bits 0-7 = GPIO 0-7
bits 8-15 = GPIO 8-15
bits 16-23 = GPIO 16-23
bits 24-31 = GPIO 24-31
bits 0-7 = GPIO 32-39
bits 8-15 = GPIO 40-47
bits 16-22 = GPIO 48-54

This register writes 0 to the GPIO pin

0x3F200028	40		**Write 1 GPIO18 (Lab 7)**
	41	set bit n to turn OFF GPIO n	mov r1,#1
	42		lsl r1,#18
	43		str r1,[r0,#40]
0x3F20002C	**44**		
	45	set bit n to turn OFF GPIO 32+n	
	46		
	47		
0x3F200030	48		

bits 0-7 = GPIO 0-7
bits 8-15 = GPIO 8-15
bits 16-23 = GPIO 16-23
bits 24-31 = GPIO 24-31
bits 0-7 = GPIO 32-39
bits 8-15 = GPIO 40-47
bits 16-22 = GPIO 48-54

Some GPIO pins need to be sent a 0 to turn the pin on, others need a 1 to turn on.

This register contains state of the GPIO pin (if programmed to read)

0x3F200034	52	
	53	read bit n to detect GPIO n
	54	
	55	
0x3F200038	56	
	57	read bit n to detect GPIO 32+n
	58	
	59	
0x3F20004C	60	

bits 0-7 = GPIO 0-7
bits 8-15 = GPIO 8-15
bits 16-23 = GPIO 16-23
bits 24-31 = GPIO 24-31
bits 0-7 = GPIO 32-39
bits 8-15 = GPIO 40-47
bits 16-22 = GPIO 48-54

TURNING OFF THE LED (GPIO 18)

```
BASE = $FE000000
GPIO_OFFSET = $200000
mov r0,BASE
orr r0,GPIO_OFFSET
;start of GPIO
...  ;program GPIO for output, turn
;set bit 18
mov r1,#1
lsl r1,#18
;write to "set low" register
str r1,[r0,#40]
```

Remember? The GPIO has (at offset):

Offset 0 bytes: Function registers
Offset 28 bytes: Set high registers
Offset 40 bytes: Set low registers
Offset 52 bytes: Read registers.
See RPiGPIO.xlsx

INSERT DELAY WITH BUSY WAIT TIMER

- Pseudocode:
  - Program GPIO18 LED for output
  - Loop1:
    - Turn LED on (pull GPIO18 high)
    - Busy wait
    - Turn LED off (pull GPIO18 low)
    - Busy wait
  - branch to loop1

SUMMARY

- To make LED flash we turn LED on an off with inserted time delays
- Pulling a GPIO register pin low follows same pattern as pulling high
  - Write “1” to the appropriate position in the “off” register
  - Everything else is the same!
- Busy wait time needs a comparison and a loop:
  - Next lecture we discuss how to do this!