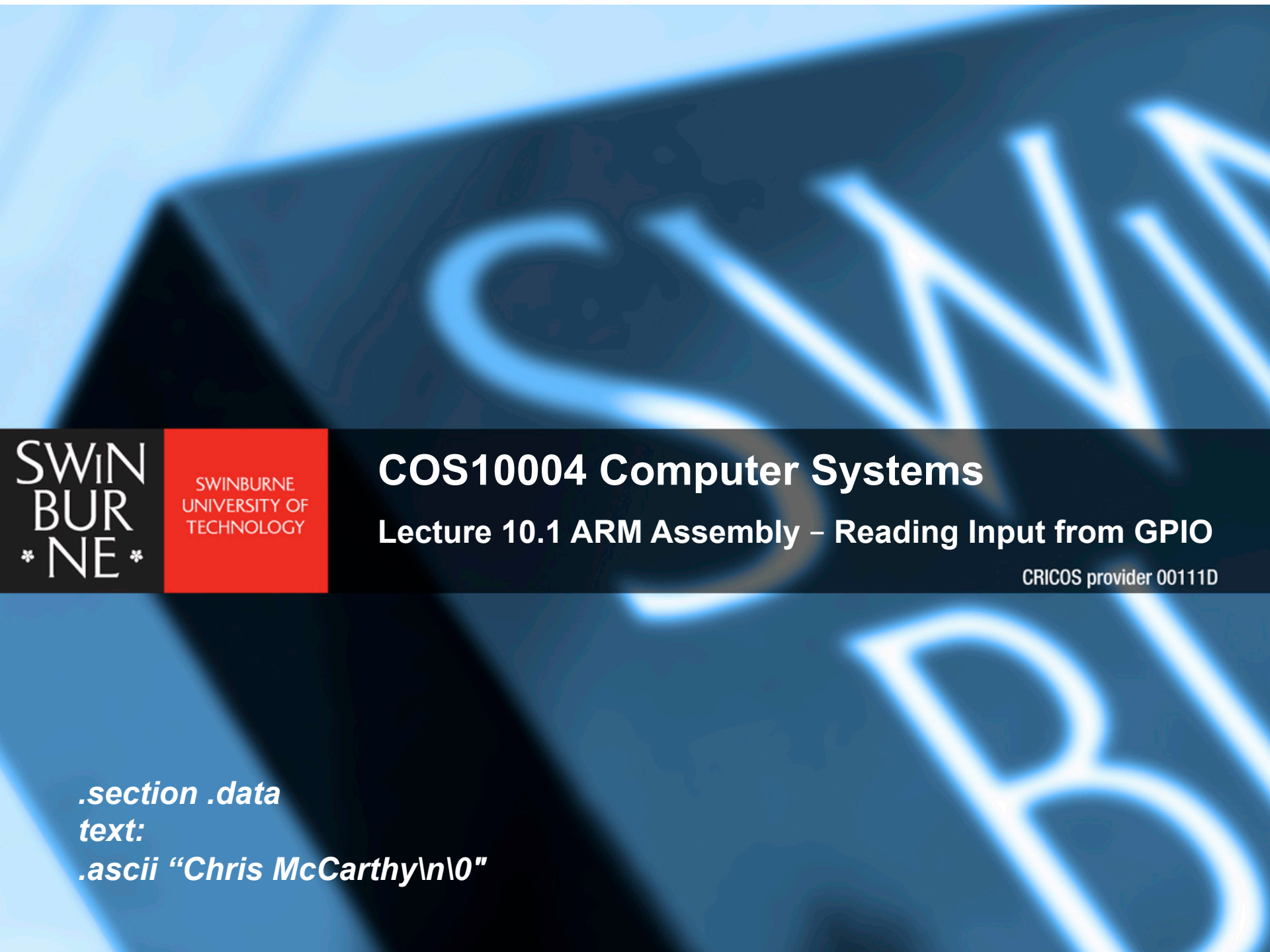


COS10004 Computer Systems

Lecture 10.1 ARM Assembly – Reading Input from GPIO

CRICOS provider 00111D

```
.section .data
text:
.ascii “Chris McCarthy\n\0"
```

Stock Take – what have we done ?

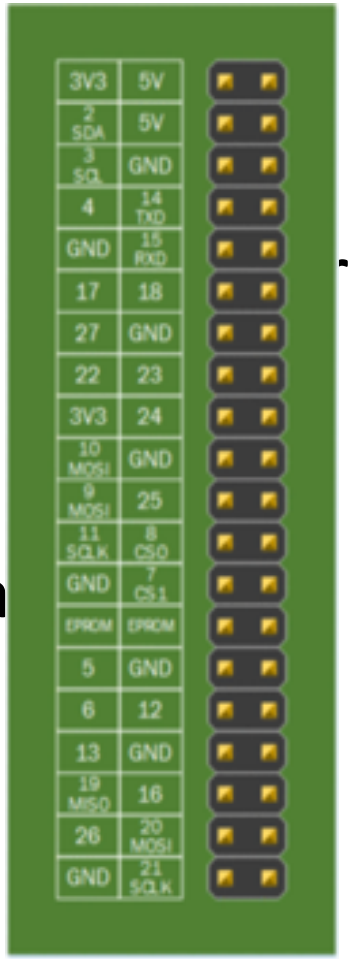
- ARM Assembly basics
- Learnt quite a few ARM Assembly instructions!
- Writing to GPIO pins to flash LEDs
- Reading from timer register
- Functions and ABI
- Stack
- Recursion

Stock Take – what's left?

- Core material:
  - Reading from GPIO pins (input!)
  - Arrays
- Optional material (but possibly very useful for assignment 2)
  - Writing to screen in bare assembly
  - ARM assembly in an Operating System

STOCK TAKE – WHAT'S LEFT?

- Core material:
  - Reading from GPIO pins (input!)
  - Arrays
- Optional material (but possibly very assignment 2)
  - Writing to screen in bare assembly
  - ARM assembly in an Operating System



Binary input

- We can read the state of a GPIO pin by:

Programming the GPIO for input.

In a loop:

Applying a voltage (0 or 3.3V) to the appropriate header pin

Read GPIOs

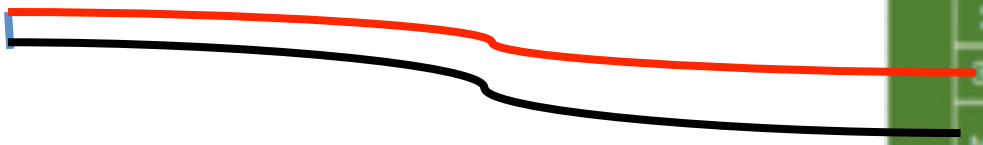
Test the state of the appropriate bit

Branch if the bit is set (or 0)

End loop

The wiring

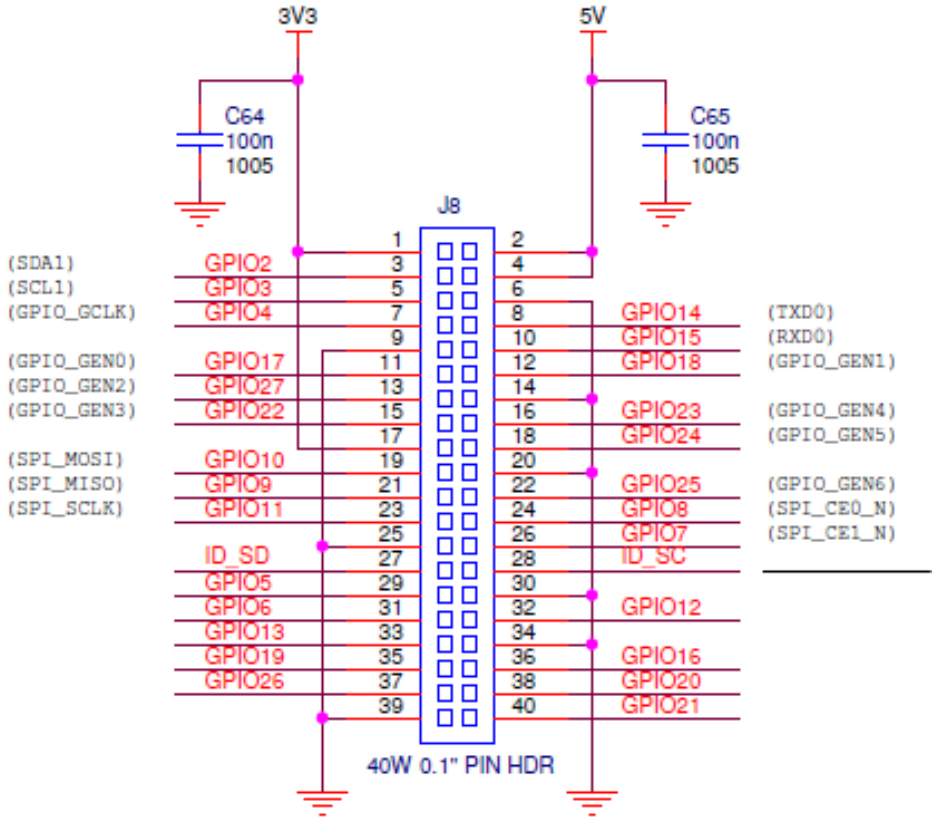
- Connect two wires to Header Pins 17 and 19.



- When we touch them together, GPIO10 is pulled high.
- Detect this in the code and change which LED flashes.

So Which GPIO register ?

- So the pin to read from is accessed via GPIO10.
- When the wires touch, GPIO10 will be pulled high: “1”
- But how do we access this to check ?



GPIO EXPANSION

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

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

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

This block of registers is for reading input. We can test a specific bit to see if it is “on” or “off”

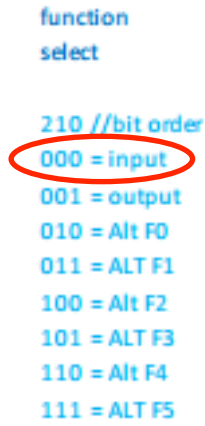
Programming for input

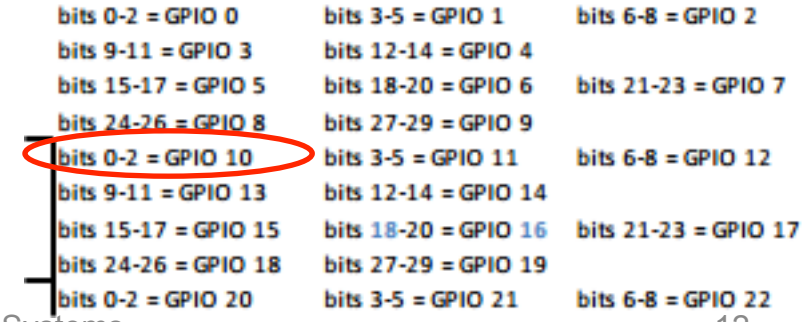
- To program output, write 001 to address in the program register.
- To program input, write 000 to address in the program register.
  - If we just ***lsl*** a 0 we set all of the other bits to 0.
  - Breaks code for other GPIOs (input and output). Need to be a bit smarter.
- Instead, we can clear a specific bit in a register using **`bic`**

Bit Clear

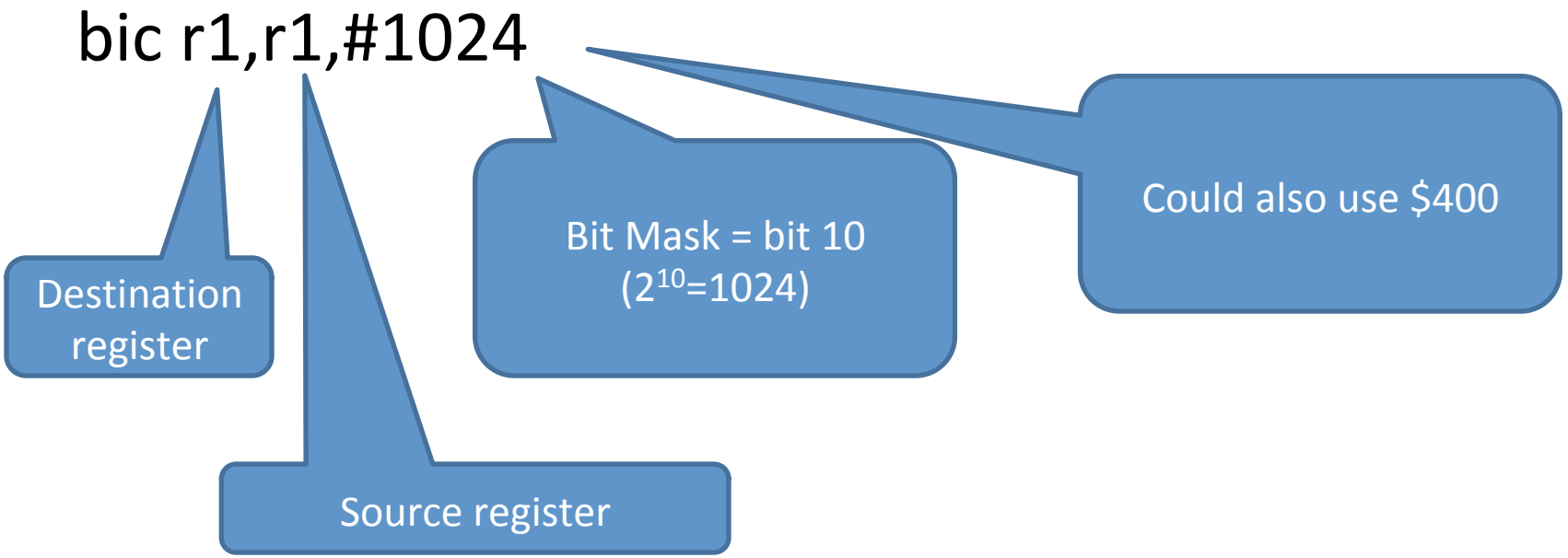
READING GPIO10 PART 1

```
BASE = $FE000000 ; Use $3F000000 for 2
GPIO_OFFSET = $200000
mov r0,BASE
orr r0,GPIO_OFFSET ;Base address of GPIO
;read the relevant function register
ldr r1,[r0,#4] ;read function register for
GPIO 10 - 19
;clear the 3 bits for GPIO10
bic r1,r1,#7  ;bit clear
str r1,[r0,#4]
```





Bit Clear



- Clear bits in register 1 if they are set in bit mask
  - R1= r1 AND NOT 1024

READING GPIO10 PART 2

```
;set up outputs
ldr r10,[r0,#4];LED1
orr r10, $1000000 ;set bit 24
str r10,[r0,#4];GPIO18 output

ldr r2,[r0,#16];green LED
orr r2, $200;set bit 9
str r2,[r0,#16];GPIO23 output
```

0x1000000 = 2^{24}

0x200= 2^{9}

READING GPIO10 PART 3

```
;set up registers with bits
set for on, off and input registers
;activate LED 1
mov r2,#1
lsl r2,#18  ;bit 18 to write to GPIO18
;disable LED 2
mov r10,#1
lsl r10,#23   ;bit 23 to write to GPIO23
```

READING GPIO10 PART 4

```
;read first block of GPIOs
ldr r9,[r0,#52] ;read gpios 0-31
tst r9,#1024  ;use tst to check bit 10
bne led2 ;if bit 10 set goto LED 2
;else set LED 1 (below)
str r2,[r0,#28] ;Turn on LED 1
b cont  ;end of if
led2:
str r10,[r0,#28] ;Turn on LED 2
cont:
```

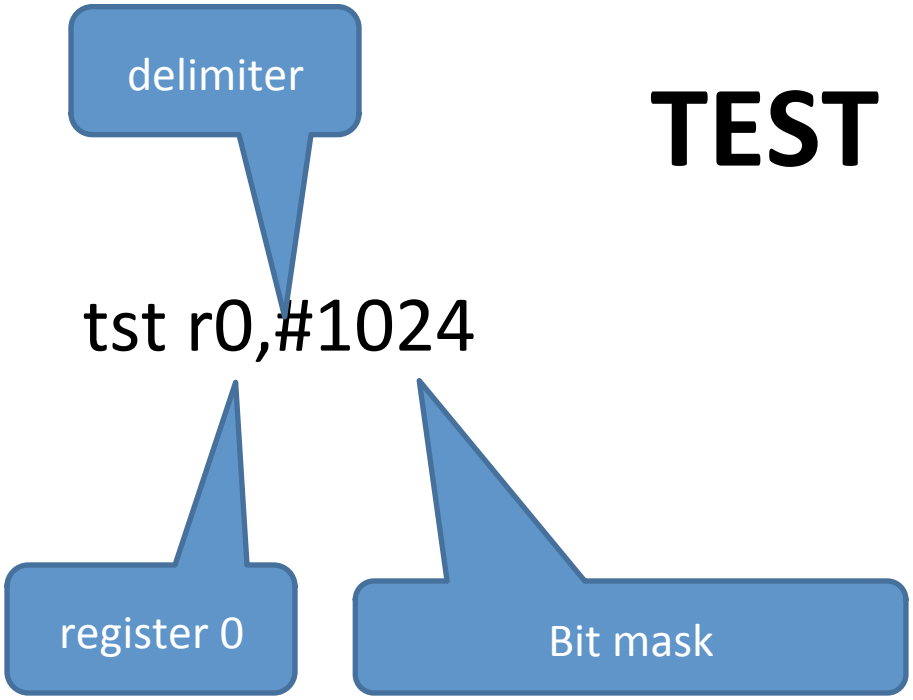
READING GPIO10 PART 4

```
;read first block of GPIO
ldr r9,[r0,#52] ;read gpi
tst r9,#1024  ;use tst to
bne led2 ;if bit 10 set goto LED 2
;else set LED 1 (below)
str r2,[r0,#28] ;Turn on
b cont  ;end of if
led2:
str r10,[r0,#28] ;Turn on
cont:
```

GPIO10 is in the first block of input registers

Check if bit 10 ==1 ($1024=2^{10}$)

TEST BIT



- Update APSR register r0 AND bitmask
- r0 AND 1024 (but does not write to register r0)

READING GPIO10 PART 4B

```
;read first block of GPIOs
ldr r9,[r0,#52] ;read gpios 0-31
tst r9,#1024  ;use tst to check bit 10
bne led2 ;if bit 10 set
;else set LED 1
str r2,[r0,#28] ;Turn on LED 1
b cont  ;end of if
led2:
str r10,[r0,#28] ;Turn on LED 2
cont:
```

If != 0 goto RED

Else continue

And then skip to end of IF

This is how we do an If…else in a ASM.

FULL CODE WALK THROUGH

SUMMARY

- We can read from GPIO pins in much the same way as we write:
  - Set up the function register by clearing the 3 bits associated with the GPIO pin
    - Use bic
  - Read the correct register from GPIO read block (starting at memory offset #54) and check the correct bit
    - Use tst
- You can use ORR to set multiple bits in a GPIO register (or any register for that matter!
  - that way you don't overwrite previously set bits