ESET 349 – Microcontroller Architecture

Port Programming

Dr. Muhammad Faeyz Karim

Digital Input & Output

MSP432P401R has 100 pins

- P1-P10 and PJ ports

- Each Port has a group of up to 8 Pins at a maximum
- Each Port has a dedicated Base Address in the GPIO memory area
  - From the base address, each port has multiple bytes for specific tasks for the port
  - Each of these bytes is called a **REGISTER** (8-bit GPIO register)
- At the beginning of a program, these pins need to be configured as **Input** or **Output** pin
- The digital output pin can provide a logic HIGH or logic LOW from the pin
- The digital input pin can read a logic HIGH or LOW from another device (button, sensor, etc.)
- HIGH is 3.3 V, LOW is 0 V

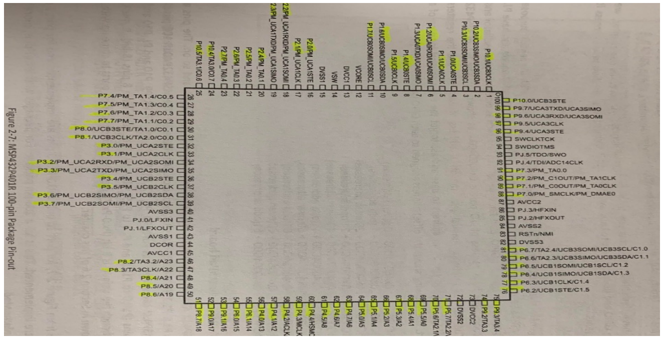
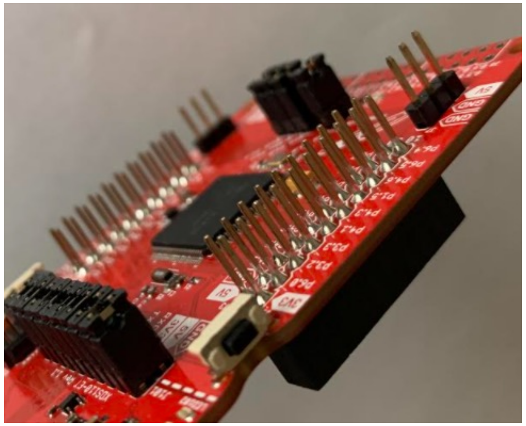


Figure 2-7: MSP432P401R 100-pin Package Pin-out



Digital Input & Output

While programming, each pin can be configured for the following services

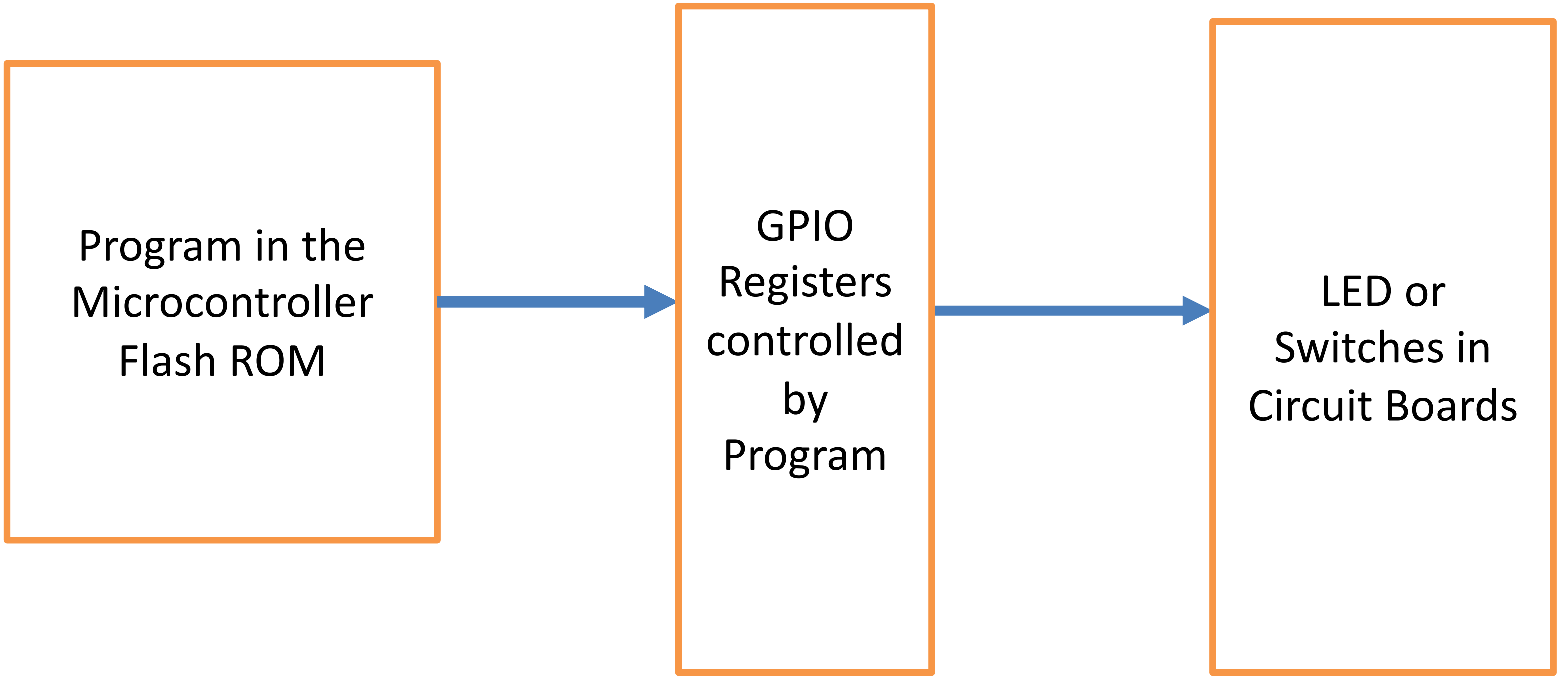
1. General Purpose I/O
2. UART, SPI, I2C
3. Timers
4. ADC, Comparator

The default use of a Pin is the GPIO communication

Of Multiple Registers Associated to Each Port, the following three major ones:

1. Input Register/byte (has a unique address)
2. Output Register/byte (has a unique address)
3. Data Direction Register/byte (has a unique address)

Programming Model



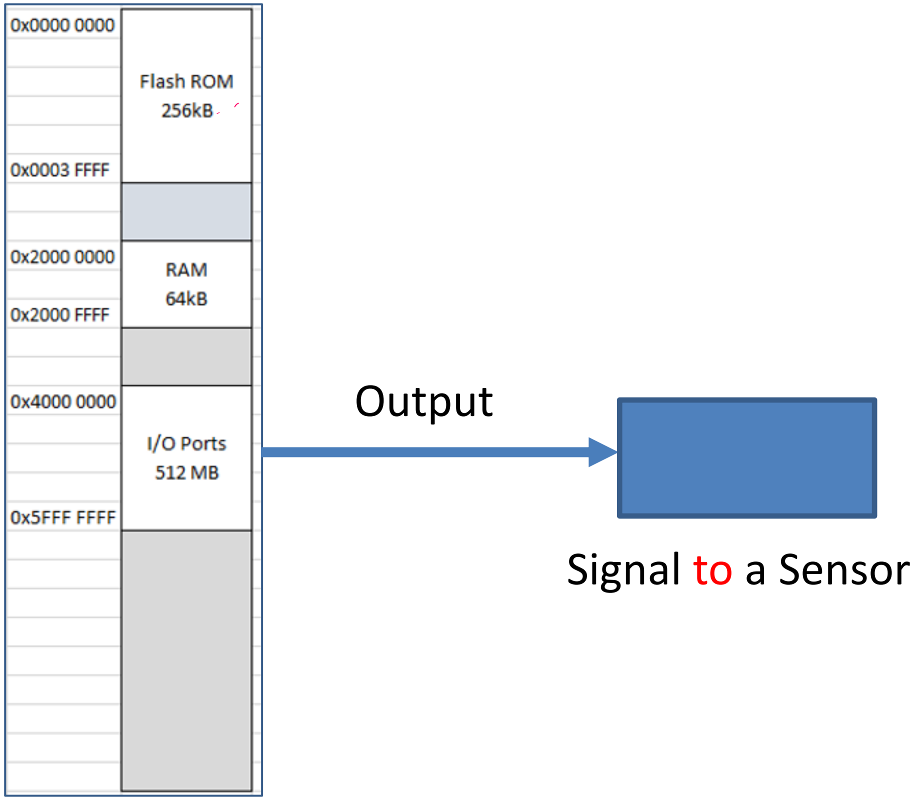
Program in the Microcontroller Flash ROM

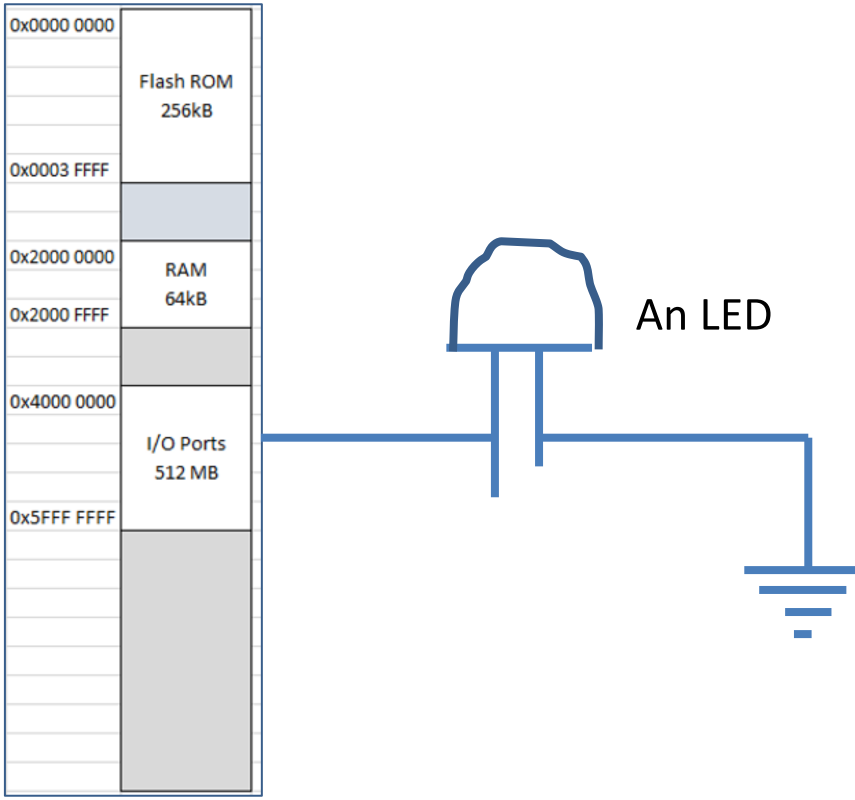
GPIO Registers controlled by Program

LED or Switches in Circuit Boards

Port Programming – I/O

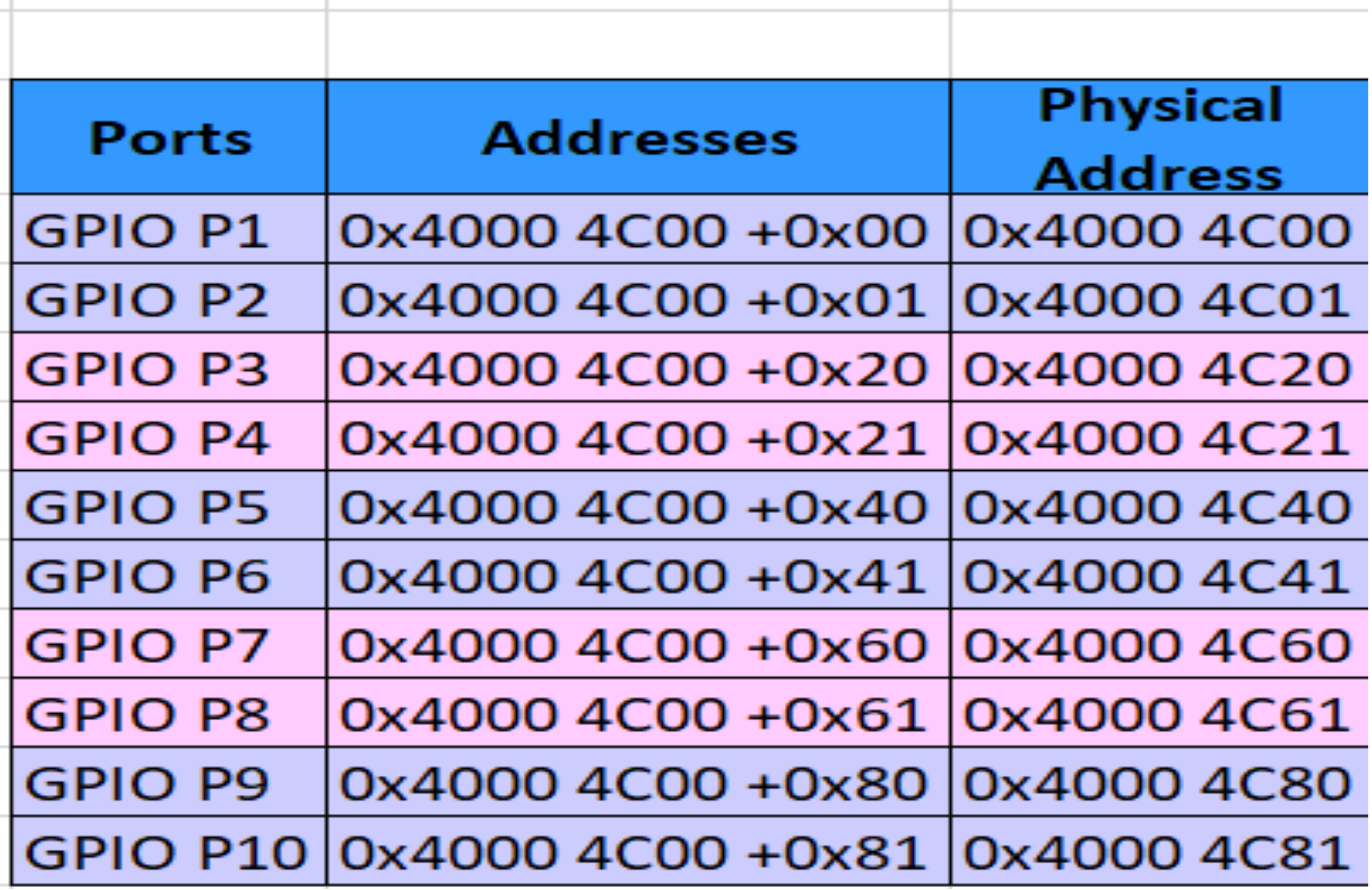
Programming to an Output Port





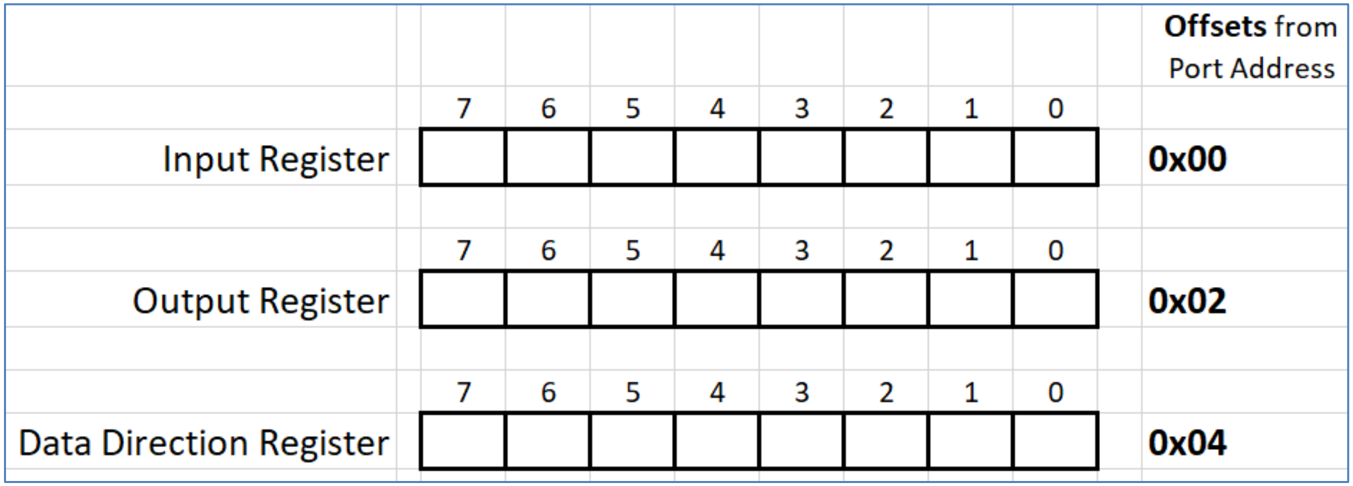
To protect the LED, make sure to add a resistor in series (220 or 330 Ohm for example)

Port Address



Ports	Addresses	Physical Address
GPIO P1	0x4000 4C00 +0x00	0x4000 4C00
GPIO P2	0x4000 4C00 +0x01	0x4000 4C01
GPIO P3	0x4000 4C00 +0x20	0x4000 4C20
GPIO P4	0x4000 4C00 +0x21	0x4000 4C21
GPIO P5	0x4000 4C00 +0x40	0x4000 4C40
GPIO P6	0x4000 4C00 +0x41	0x4000 4C41
GPIO P7	0x4000 4C00 +0x60	0x4000 4C60
GPIO P8	0x4000 4C00 +0x61	0x4000 4C61
GPIO P9	0x4000 4C00 +0x80	0x4000 4C80
GPIO P10	0x4000 4C00 +0x81	0x4000 4C81

Offsets of THREE major Registers



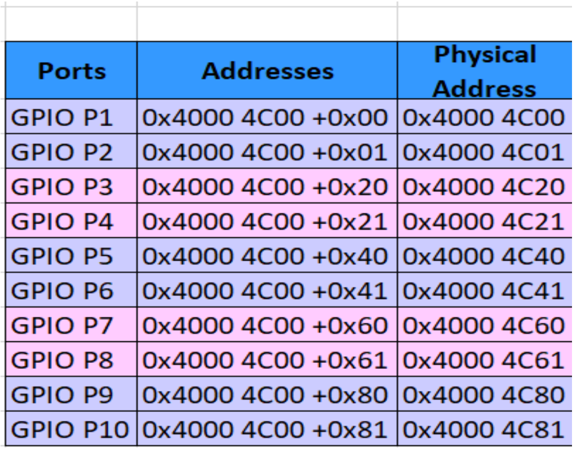
	7	6	5	4	3	2	1	0	**Offsets** from Port Address
Input Register									**0x00**
Output Register									**0x02**
Data Direction Register									**0x04**

Exercise 1: Find the absolute address of the Data Direction Register of Port 7?

```
LDR r0, =0x40004C00            ; Base address of GPIO registers
ADD r0, #0x60                  ; R0 has the address of Port 7
ADD r1, r0, #0x04              ; R1 has the absolute address of DDIR
```

Exercise 2: What is the absolute address of the OUTPUT byte of Port 2?

```
LDR R0, =0x40004C00            ; Base address of the GPIO registers
ADD r0, #0x01                  ; Address of Port 2
ADD r2, r0, #0x02              ; R2 has the absolute address of OUTPUT register
```



Ports	**Addresses**	**Physical Address**
GPIO P1	0x4000 4C00 +0x00	0x4000 4C00
GPIO P2	0x4000 4C00 +0x01	0x4000 4C01
GPIO P3	0x4000 4C00 +0x20	0x4000 4C20
GPIO P4	0x4000 4C00 +0x21	0x4000 4C21
GPIO P5	0x4000 4C00 +0x40	0x4000 4C40
GPIO P6	0x4000 4C00 +0x41	0x4000 4C41
GPIO P7	0x4000 4C00 +0x60	0x4000 4C60
GPIO P8	0x4000 4C00 +0x61	0x4000 4C61
GPIO P9	0x4000 4C00 +0x80	0x4000 4C80
GPIO P10	0x4000 4C00 +0x81	0x4000 4C81

Configuring an OUTPUT/INPUT Pin

- A Pin can be configured as INPUT or OUTPUT using a particular Register (DDIR byte) of a Port.
- If a bit has a value of 1, the Pin is responsible for outputting a High signal to OUTPUT register, a Low signal, 0 to the same OUTPUT register
- To read a signal we read the specific bit of the INPUT register

- What is the byte to configure P3.4 and P3.0 as output pins?

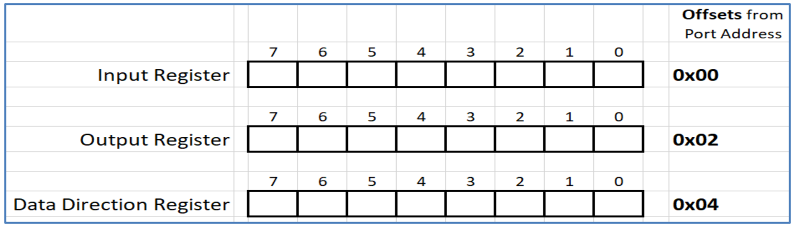
7 6 5 4 3 2 1 0 (bit locations)

0 0 0 1 0 0 0 1 (1's indicate that the connected Pins responsible for outputting signals)

The Data Direction byte in hex format is: **0x11**

○ The remaining Pins will remain as if they are configured as INPUT pins

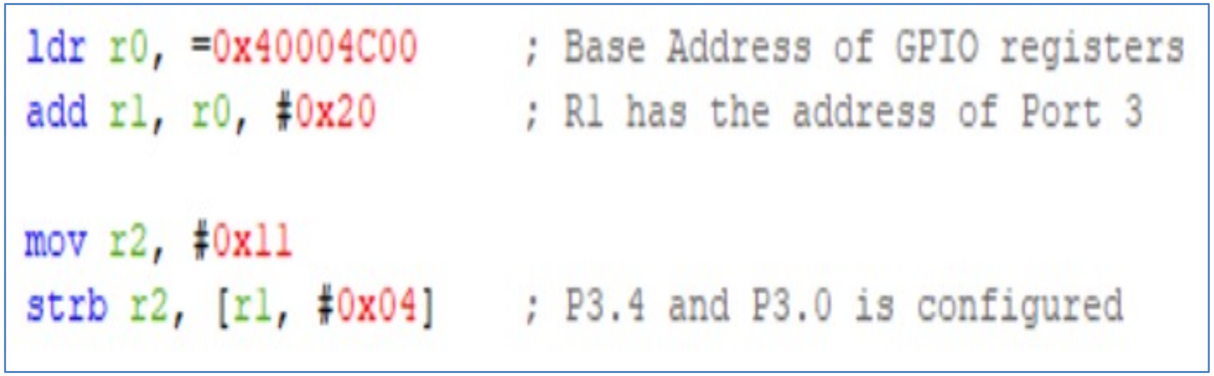
○ This byte is called the DATA Direction byte.



									Offsets from Port Address
	7	6	5	4	3	2	1	0	
Input Register									0x00
	7	6	5	4	3	2	1	0	
Output Register									0x02
	7	6	5	4	3	2	1	0	
Data Direction Register									0x04

Configuring an INPUT/OUTPUT Pin

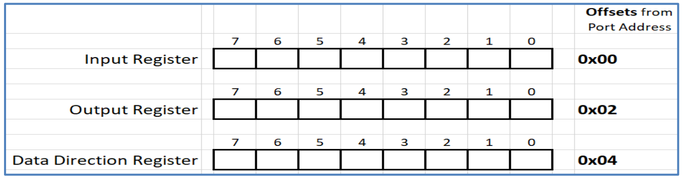
- The Data Direction byte needs to be stored in the Data Direction Register in the Memory
- The offset of Data Direction Register is **0x04** from the Base or Port Address that the Pin(s) belongs to.



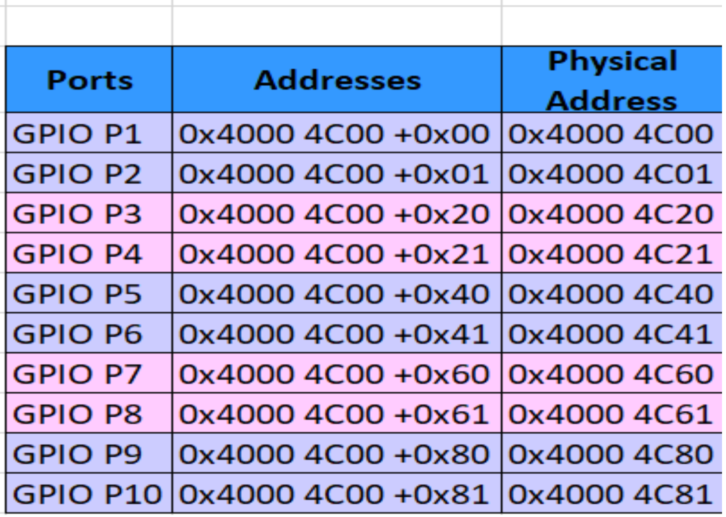
```
ldr r0, =0x40004C00       ; Base Address of GPIO registers
add r1, r0, #0x20         ; R1 has the address of Port 3

mov r2, #0x11
strb r2, [r1, #0x04]      ; P3.4 and P3.0 is configured
```

Exercise: Configure P5.7, P5.3, and P5.1 as Output Pins

```
LDR r0, =0x40004C00
ADD r0, #0x40          ; R0 now points to Port 5
MOV r1, #0x8A          ; Data Direction byte
STRB r1, [r0, #0x04]
```



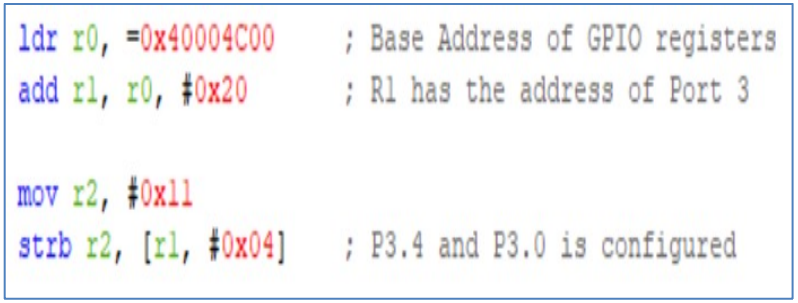
	7	6	5	4	3	2	1	0	Offsets from Port Address
Input Register									0x00
Output Register									0x02
Data Direction Register									0x04



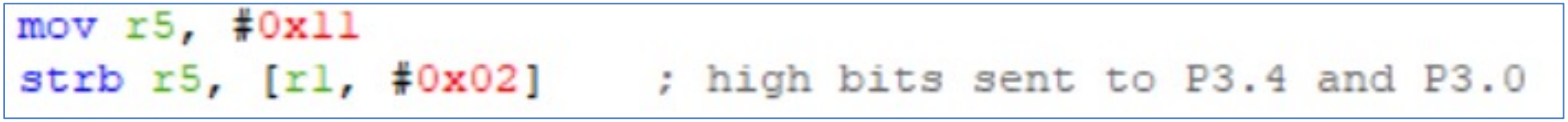
Ports	Addresses	Physical Address
GPIO P1	0x4000 4C00 +0x00	0x4000 4C00
GPIO P2	0x4000 4C00 +0x01	0x4000 4C01
GPIO P3	0x4000 4C00 +0x20	0x4000 4C20
GPIO P4	0x4000 4C00 +0x21	0x4000 4C21
GPIO P5	0x4000 4C00 +0x40	0x4000 4C40
GPIO P6	0x4000 4C00 +0x41	0x4000 4C41
GPIO P7	0x4000 4C00 +0x60	0x4000 4C60
GPIO P8	0x4000 4C00 +0x61	0x4000 4C61
GPIO P9	0x4000 4C00 +0x80	0x4000 4C80
GPIO P10	0x4000 4C00 +0x81	0x4000 4C81

Sending Signal to

- To send high signals to P3.4 and P3.0 which, in turn, connected to two LEDs, for example:
  - The byte is **0x11**
- The offset of OUTPUT register is **0x02**
- Therefore, the statements should be



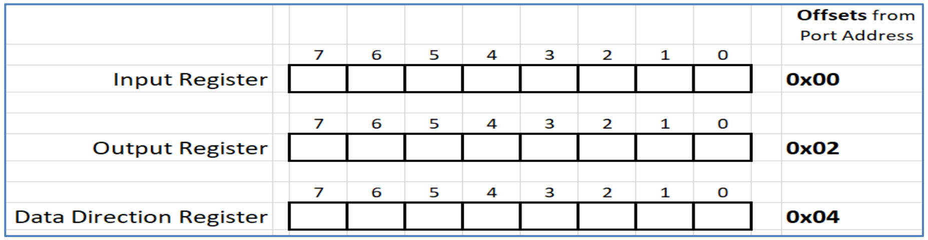
```
ldr r0, =0x40004C00        ; Base Address of GPIO registers
add r1, r0, #0x20          ; R1 has the address of Port 3

mov r2, #0x11
strb r2, [r1, #0x04]       ; P3.4 and P3.0 is configured
```



```
mov r5, #0x11
strb r5, [r1, #0x02]      ; high bits sent to P3.4 and P3.0
```

Exercise: Send high bits to P5.7, P5.3, and P5.1. Assume they are configured as OUTPUT pins

The output byte = 0x8A



	7	6	5	4	3	2	1	0	Offsets from Port Address
Input Register									0x00
Output Register									0x02
Data Direction Register									0x04

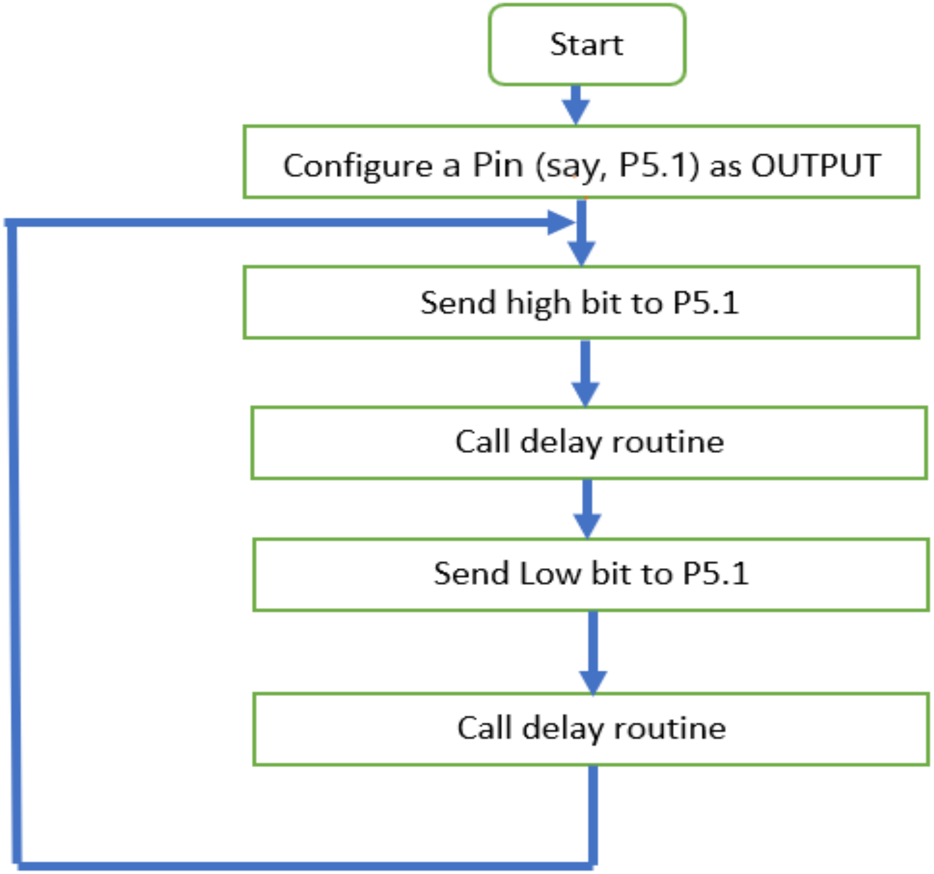
Toggling an LED

- Mechanism of Toggling an LED in a breadboard Circuit

  1. Send a High signal
  2. Wait for some time to visualize the light emitting from LED
  3. Send a Low signal
  4. Repeat the sequence from step 1

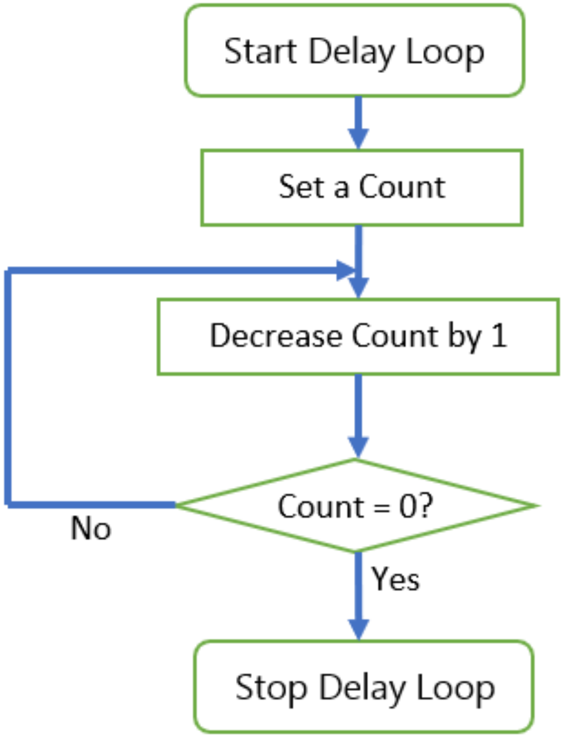
Toggling LED

Notice that the loop is Infinite.



Flowchart for a Single Loop Delay Routine

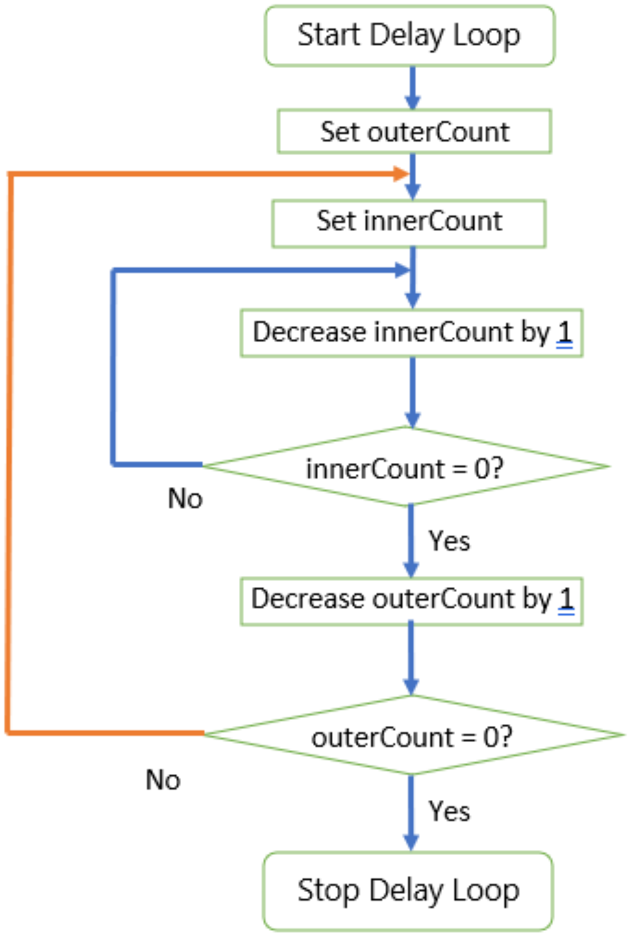
Single Loop Delay Routine



A Flowchart for a Single Loop Delay Routine

Two Loop Delay Flowchart

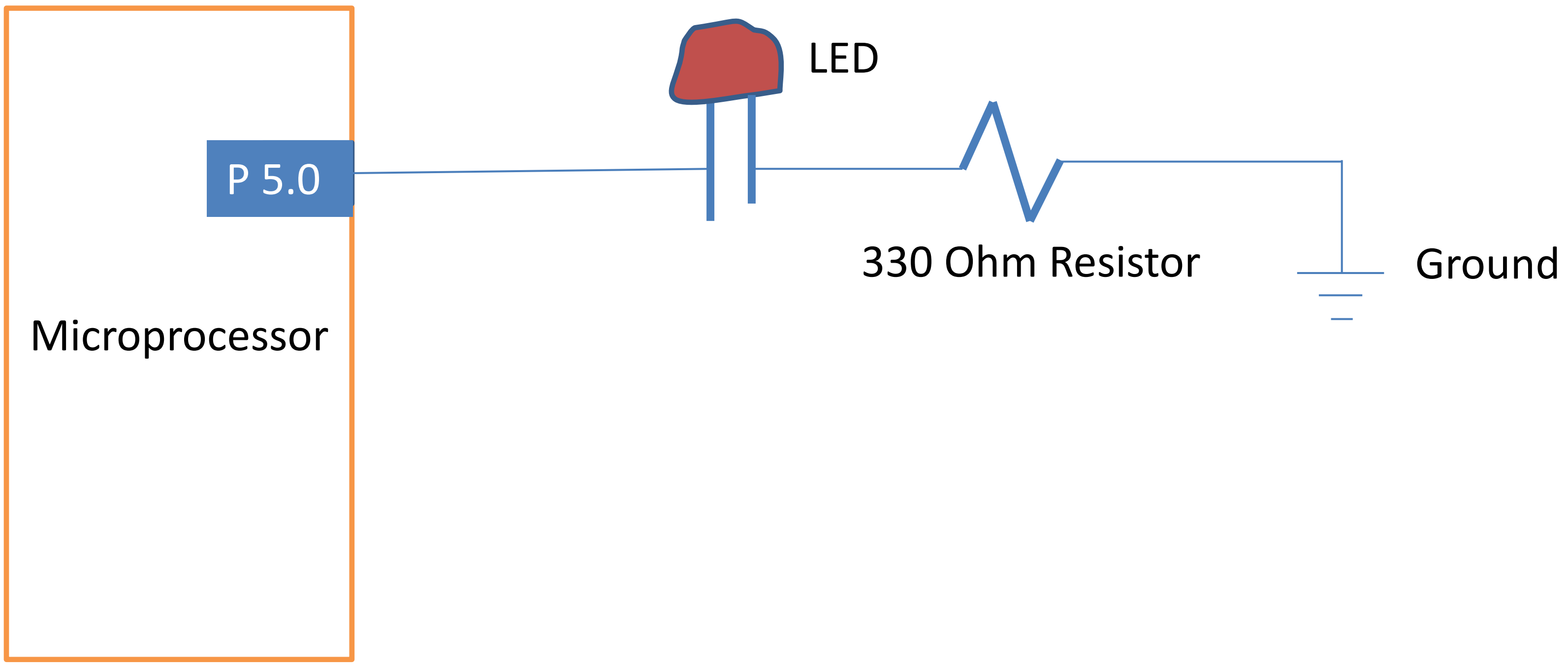
A Flowchart for a Two Loop Delay Routine



A Flowchart for a Two Loop Delay Routine

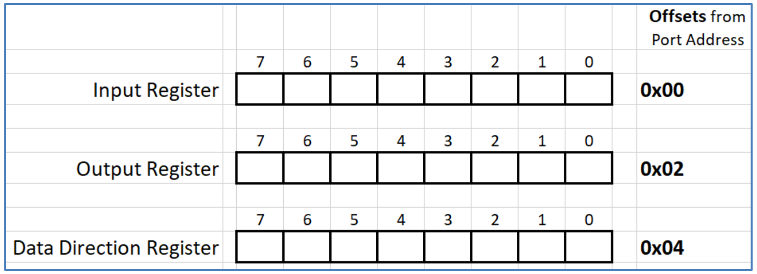
A simple circuit Diagram for LED in Breadboard

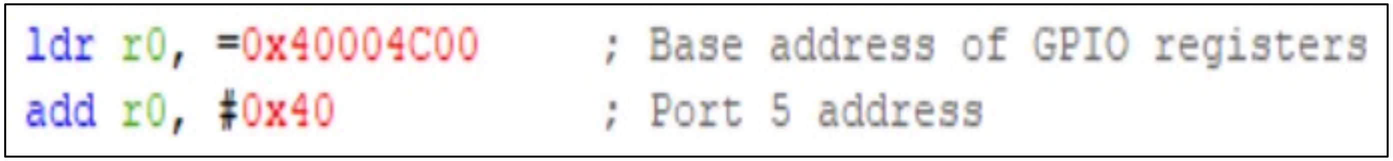
LED circuit diagram



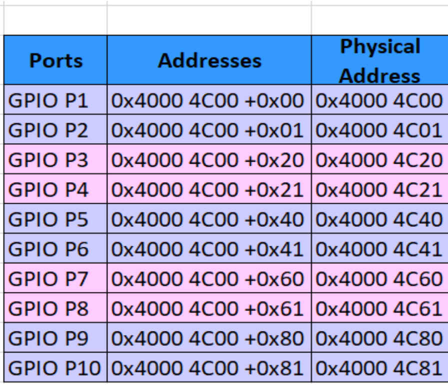
Configuring P5.0 as OUTPUT Pin

- Port 5 address Configured
  - Offset of Port 5 is 0x40



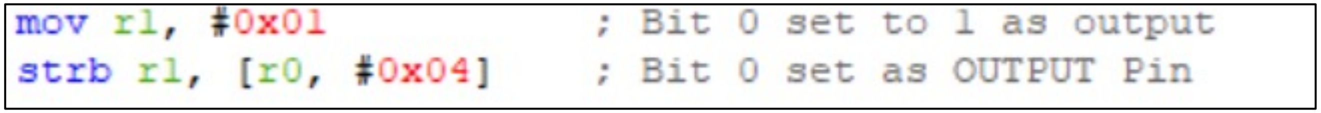


```
ldr r0, =0x40004C00       ; Base address of GPIO registers
add r0, #0x40             ; Port 5 address
```



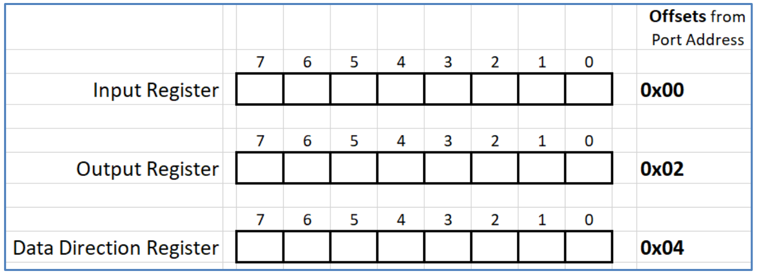
Ports	Addresses	Physical Address
GPIO P1	0x4000 4C00 +0x00	0x4000 4C00
GPIO P2	0x4000 4C00 +0x01	0x4000 4C01
GPIO P3	0x4000 4C00 +0x20	0x4000 4C20
GPIO P4	0x4000 4C00 +0x21	0x4000 4C21
GPIO P5	0x4000 4C00 +0x40	0x4000 4C40
GPIO P6	0x4000 4C00 +0x41	0x4000 4C41
GPIO P7	0x4000 4C00 +0x60	0x4000 4C60
GPIO P8	0x4000 4C00 +0x61	0x4000 4C61
GPIO P9	0x4000 4C00 +0x80	0x4000 4C80
GPIO P10	0x4000 4C00 +0x81	0x4000 4C81

- Pin 0 Configured as OUTPUT
  - Offset of Direction Register is 0x04 from the Port Address



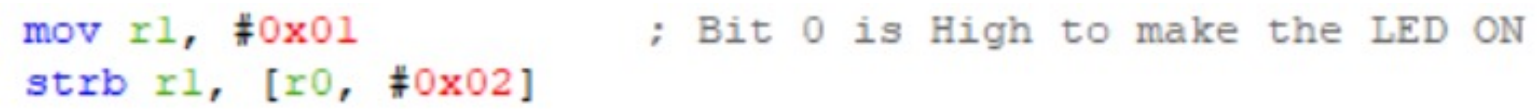
```
mov rl, #0x01            ; Bit 0 set to 1 as output
strb rl, [r0, #0x04]     ; Bit 0 set as OUTPUT Pin
```

Sending High and Low Signals to LED



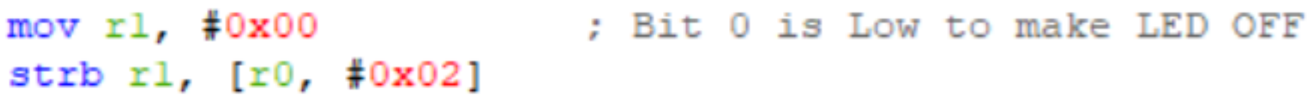
	7	6	5	4	3	2	1	0	**Offsets** from Port Address
Input Register									**0x00**
	7	6	5	4	3	2	1	0	
Output Register									**0x02**
	7	6	5	4	3	2	1	0	
Data Direction Register									**0x04**

- High bit Signal to P5.0
  - Offset of Output Register is 0x02

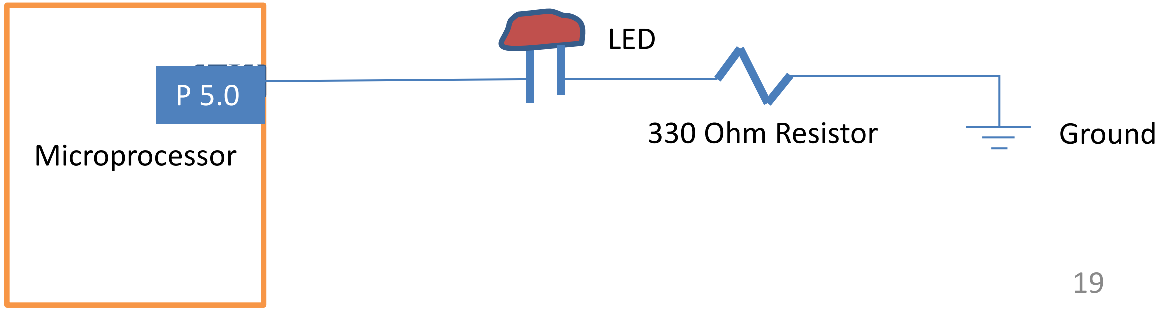


```
mov rl, #0x01                ; Bit 0 is High to make the LED ON
strb rl, [r0, #0x02]
```

- Low bit Signal to P5.0
  - Offset of Output Register is 0x02

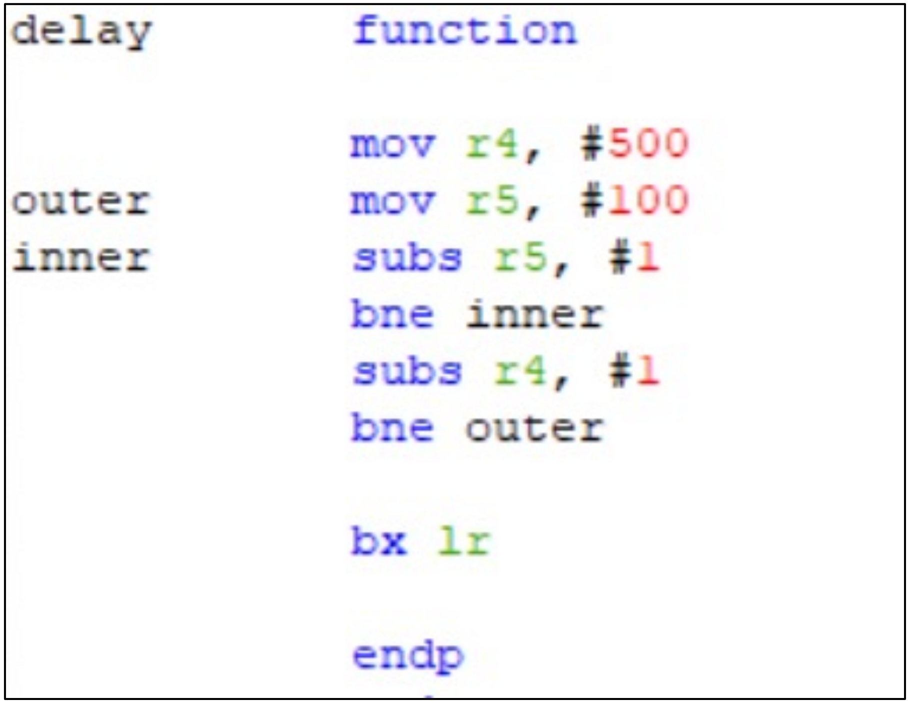


```
mov rl, #0x00              ; Bit 0 is Low to make LED OFF
strb rl, [r0, #0x02]
```



Delay Routine

- Two Loop Delay Routine



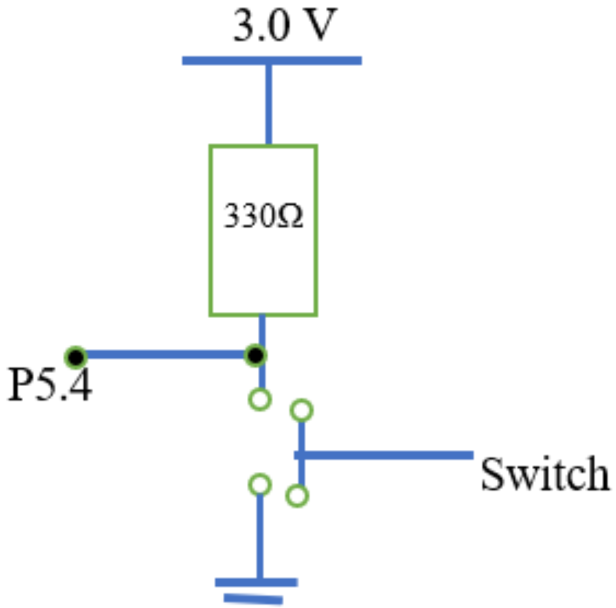
```
delay           function

                mov r4, #500
outer           mov r5, #100
inner           subs r5, #1
                bne inner
                subs r4, #1
                bne outer

                bx lr

                endp
```

INPUT Configuration

- Digital Input Configuration requires some effort for built-in input devices
- An Input is read from Input Register (GPIO) at an offset of 0x00 of the Port in question
- An Input Pin has a 0 or a 1; nothing in between for DIGITAL input
  - A push button switch can be used as an input
  - This button can be connected for Pin to have a 0 or 1 depending on how the button connects the Pin to Vcc or to ground

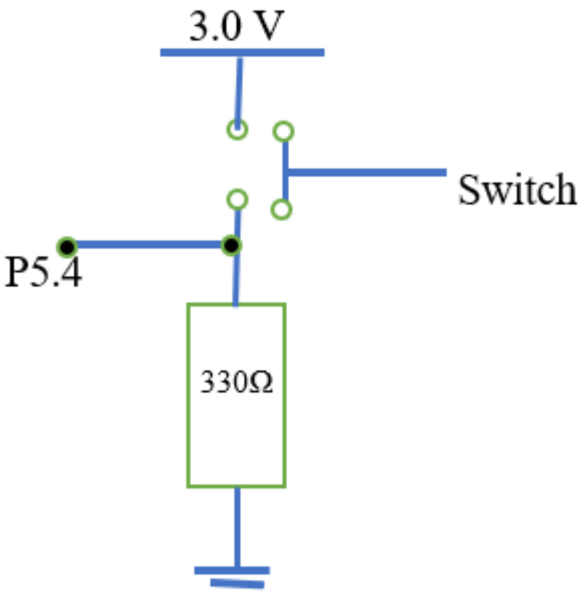
Circuit Diagram for a Pull-Up Resistor

- Pull Up Resistor connected to Input Pin P5.4
  - The default input is a 1
  - When pressed, the input is 0



Circuit Diagram of a Pull-Down Resistor

- Pull Down Resistor Connected to P5.4
  - – Unlike the Pull-up resistor, the default input is 0
  - – When the switch is pressed, the input is 1



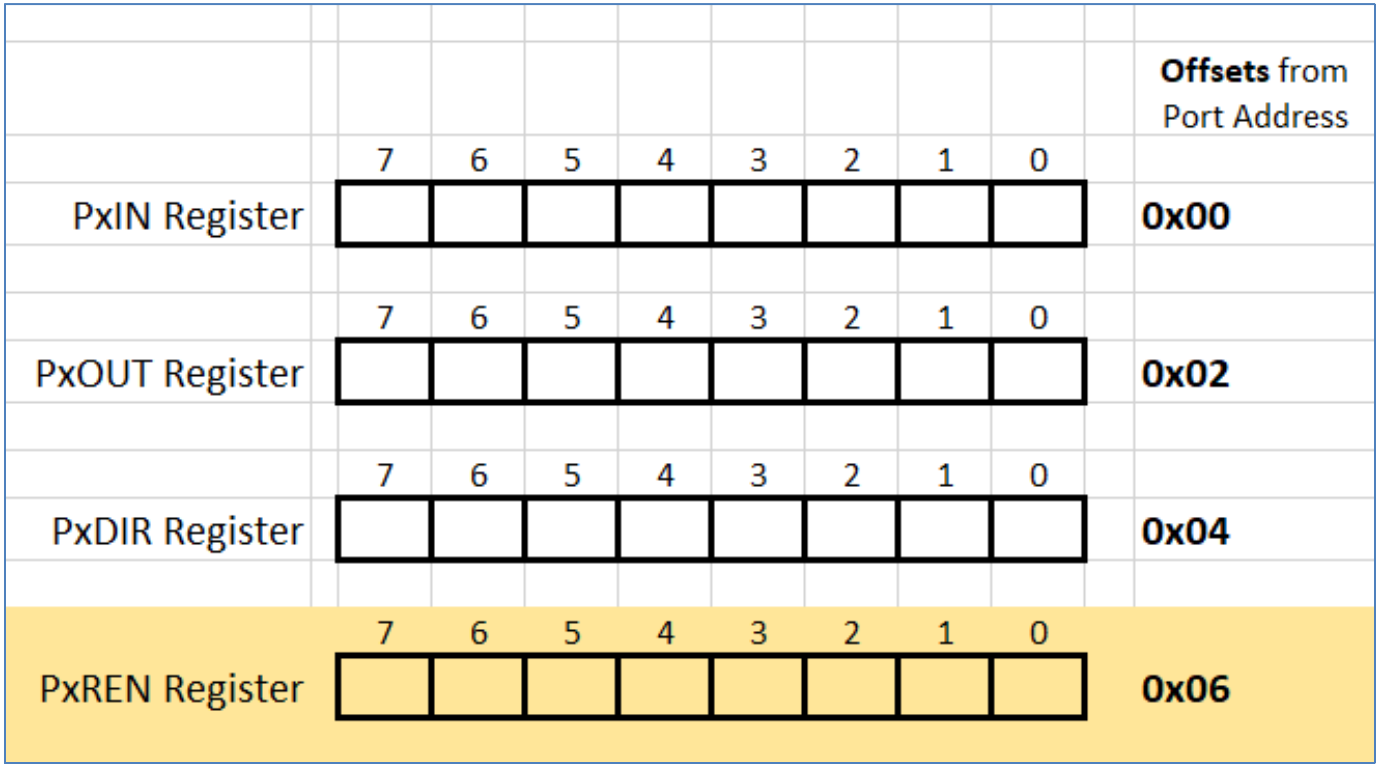
Resistors needed to Configure an Input Switch

- The REN and OUT registers are needed to configured for the built-in resistors in the LaunchPad, as
  1. The REN (Resistor Enable) enables the resistor for input Pin
     - The Offset of this Register in GPIO memory is 0x06
  2. The Pull-Up or Pull-Down Resistor is selected by the OUTPUT Register (GPIO)
     - In this case, the function of this OUT register is NOT to send signal to output Pin
     - A value 1 for an Input Pin indicates the Pull-up Resistor setup
     - A value 0 for an Input Pin indicates the Pull-Down Resistor setup
     - The offset of the Out Register is the same, 0x02
- The REN and OUT register configurations are NOT needed if the resistors are used outside the launchPad as in the breadboard.

1. The Input Register (GPIO) is for loading/reading the input value to a CPU Register
   - The Offset of Input Register is 0x00
2. The Data Direction Register is used for configuring Pin as input Pin
   - Offset is the same, 0x04

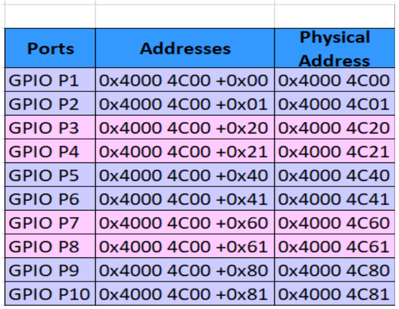
Resistor Enable - Offset

Offset of the REN Register

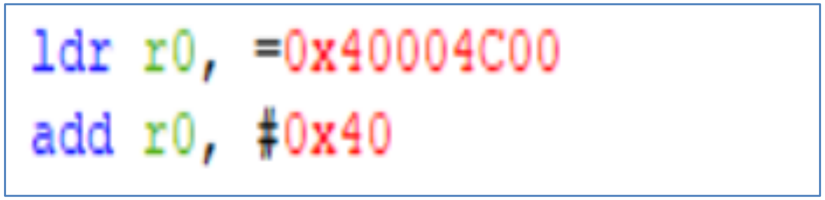


	7	6	5	4	3	2	1	0	**Offsets** from Port Address
PxIN Register									**0x00**
PxOUT Register									**0x02**
PxDIR Register									**0x04**
PxREN Register									**0x06**

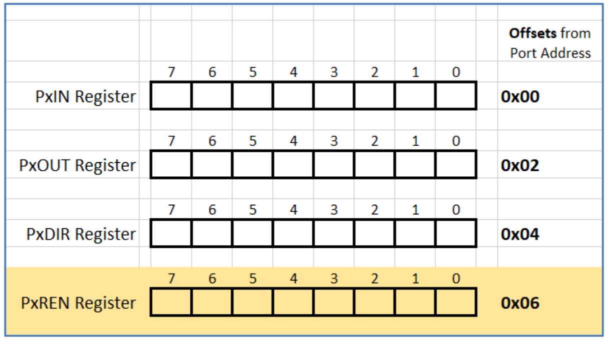
Example on Configuring an Input Pin



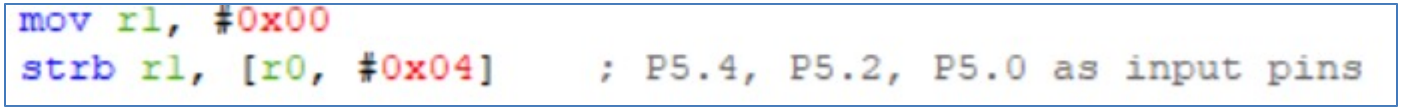
Ports	Addresses	Physical Address
GPIO P1	0x4000 4C00 +0x00	0x4000 4C00
GPIO P2	0x4000 4C00 +0x01	0x4000 4C01
GPIO P3	0x4000 4C00 +0x20	0x4000 4C20
GPIO P4	0x4000 4C00 +0x21	0x4000 4C21
GPIO P5	0x4000 4C00 +0x40	0x4000 4C40
GPIO P6	0x4000 4C00 +0x41	0x4000 4C41
GPIO P7	0x4000 4C00 +0x60	0x4000 4C60
GPIO P8	0x4000 4C00 +0x61	0x4000 4C61
GPIO P9	0x4000 4C00 +0x80	0x4000 4C80
GPIO P10	0x4000 4C00 +0x81	0x4000 4C81



```
ldr r0, =0x40004C00
add r0, #0x40
```

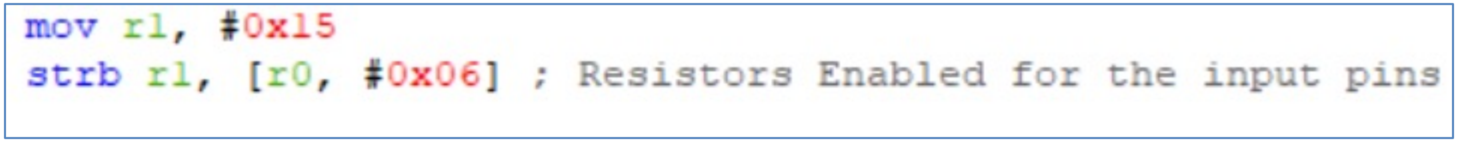


1. Configure input pins (Data Direction Register – 0x04)



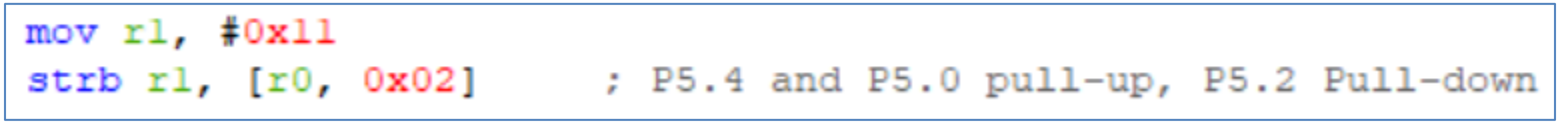
```
mov r1, #0x00
strb r1, [r0, #0x04]      ; P5.4, P5.2, P5.0 as input pins
```

2. Enable the Resistor for Pins (Resistor Register – 0x06)



```
mov r1, #0x15
strb r1, [r0, #0x06] ; Resistors Enabled for the input pins
```

3. Configure Pull-up and Pull-down Resistors (Output Register – 0x02)



```
mov r1, #0x11
strb r1, [r0, 0x02]      ; P5.4 and P5.0 pull-up, P5.2 Pull-down
```

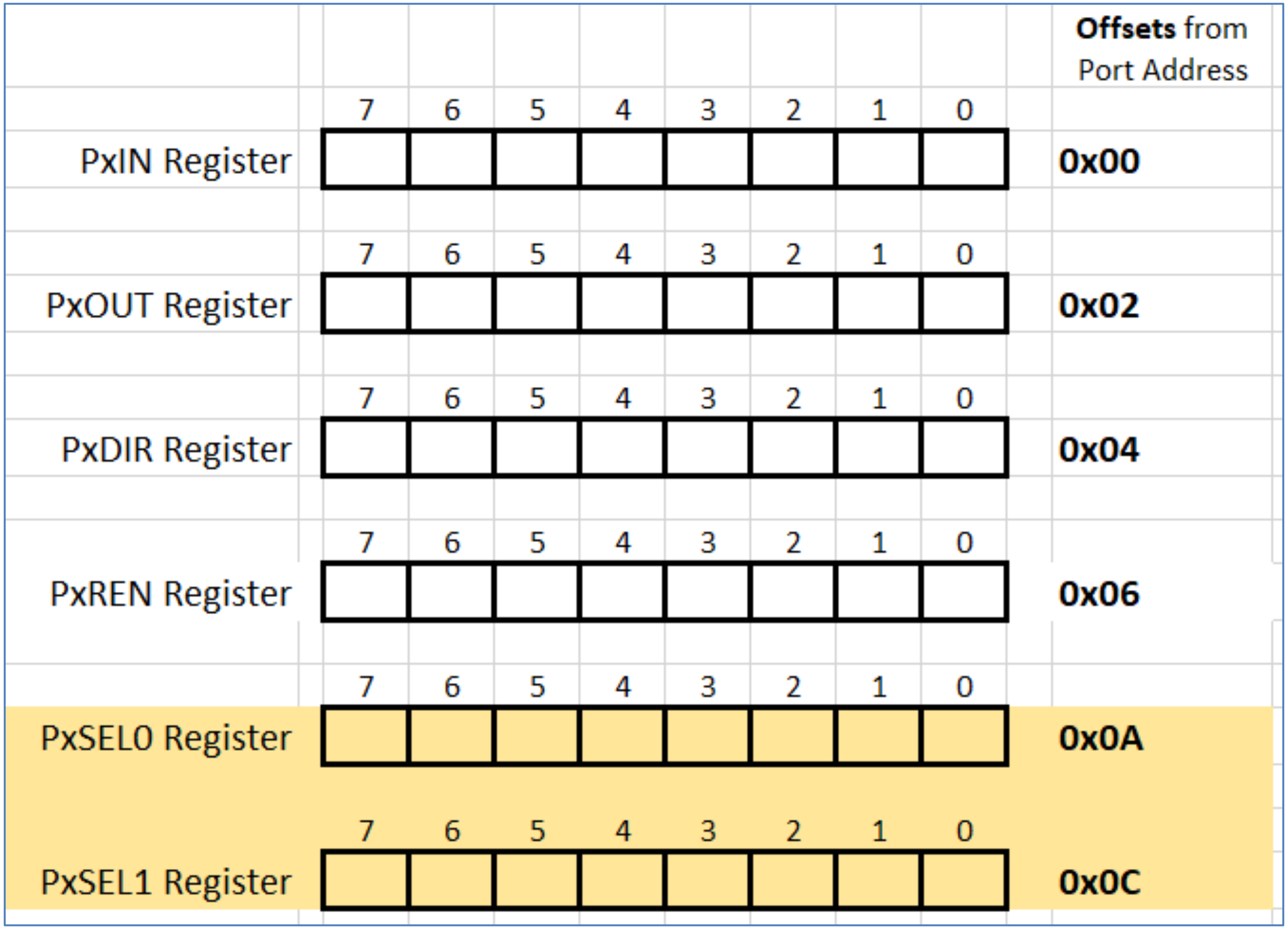
Summary – Configuration of Input Pin

Three Registers are involved

- Data Direction Register (Offset: 0x04)
- Resistor Enable Register (Offset: 0x06)
- Output Register (Offset: 0x02)

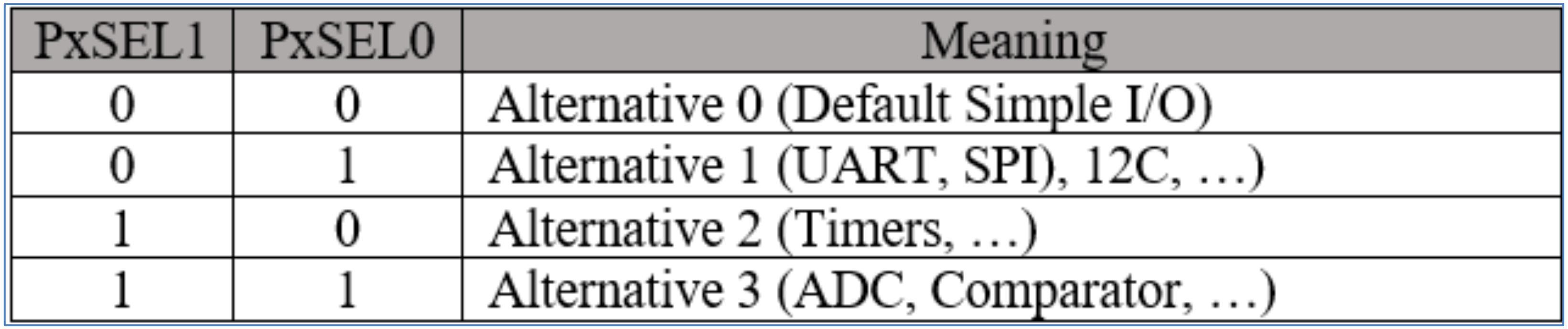
Alternative Functions of GPIO Pins

1. The Default configuration of Pins is for General Purpose Input/Output usage.
2. PxSEL0 and PxSEL1 registers are used for Alternate usages of the GPIO pins

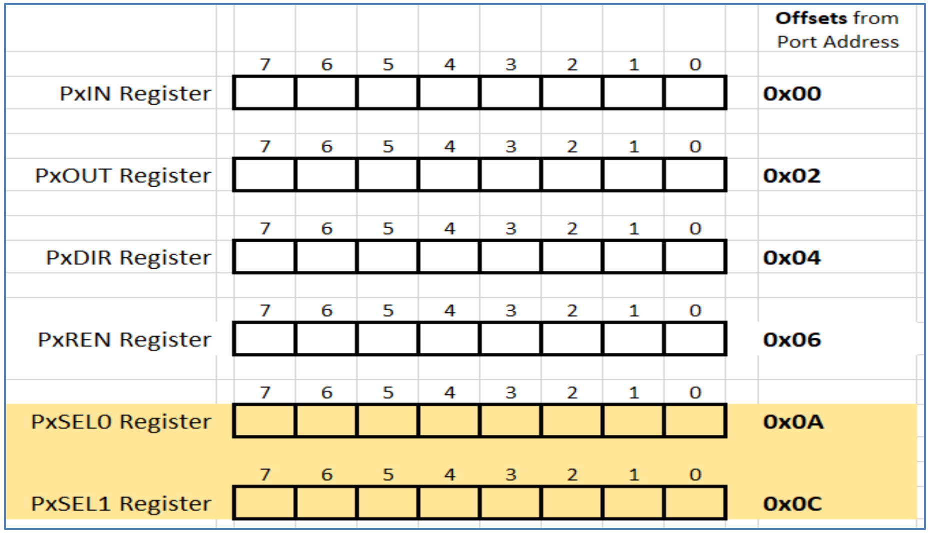


	7	6	5	4	3	2	1	0	**Offsets** from Port Address
PxIN Register									**0x00**
PxOUT Register									**0x02**
PxDIR Register									**0x04**
PxREN Register									**0x06**
PxSEL0 Register									**0x0A**
PxSEL1 Register									**0x0C**

Alternative Functions of GPIO Pins

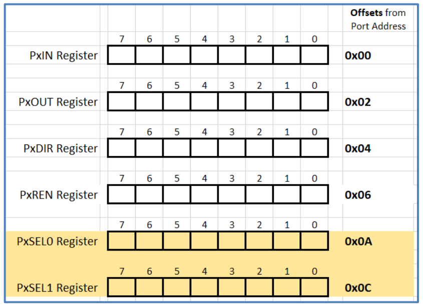


PxSEL1	PxSEL0	Meaning
0	0	Alternative 0 (Default Simple I/O)
0	1	Alternative 1 (UART, SPI), 12C, …)
1	0	Alternative 2 (Timers, …)
1	1	Alternative 3 (ADC, Comparator, …)



	7	6	5	4	3	2	1	0	**Offsets** from Port Address
PxIN Register									**0x00**
PxOUT Register									**0x02**
PxDIR Register									**0x04**
PxREN Register									**0x06**
PxSEL0 Register									**0x0A**
PxSEL1 Register									**0x0C**

Alternative Functions of GPIO Pins



	7	6	5	4	3	2	1	0	**Offsets** from Port Address
PxIN Register									0x00
PxOUT Register									0x02
PxDIR Register									0x04
PxREN Register									0x06
PxSEL0 Register									0x0A
PxSEL1 Register									0x0C

Example 1: UART0 uses P1.2 to receive data and P1.3 to transmit data. Configure these pins for UART communication

P1SEL0 byte: 0000 0100 = 0x04 for P1.2
P1SEL0 byte: 0000 1000 = 0x08 for P1.3

P1SEL0 byte for both Pins: 0000 1100 = 0x0C

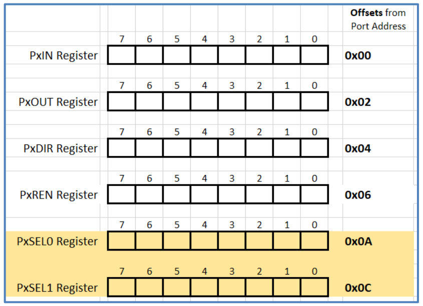
```
; Assume R1 points to the address of Port 1
LDR r1, =0x40004C00      ; Base address
add r1, #0x00            ; Port 1 address
mov r2, #0x0C
strb r2, [r1, #0x0A]          ; P1SEL0
```

P1SEL1 byte: 0000 0000 = 0x00 for P1.2
P1SEL1 byte: 0000 0000 = 0x00 for P1.3

P1SEL1 byte for both pins: 0000 0000 = 0x00

```
; Assume R1 points to the address of Port 1
ldr r1, =0x40004C00
add r1, #0x00
mov r2, #00
strb r2, [r1, #0x0C]          ; P1SEL1
```

Alternative Functions of GPIO Pins



	7	6	5	4	3	2	1	0	Offsets from Port Address
PxIN Register									0x00
PxOUT Register									0x02
PxDIR Register									0x04
PxREN Register									0x06
PxSEL0 Register									0x0A
PxSEL1 Register									0x0C

- **Example 2:** UART3 uses P9.6 to receive data and P9.7 to transmit data. Configure these pins for UART communication

P9SEL0 byte: 0100 0000 = 0x40 for P9.6
P9SEL0 byte: 1000 0000 = 0x80 for P9.7

P9SEL0 byte for both pins: 1100 0000 =0xE0

```
; Assume R1 points to the address of Port 9

LDR r1, =ox40004C00
ADD r1, #0x80               ; r1 points to Port 9
mov r2, #0xE0
strb r2, [r1, #0x0A]
```

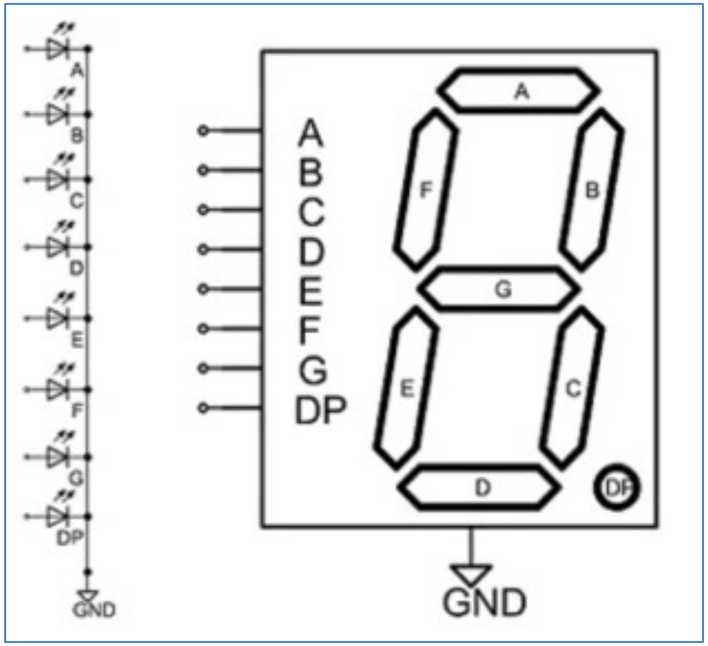
P9SEL1 byte: 0000 0000 = 0x00
P9SEL1 byte: 0000 0000 = 0x00

P9SEL1 byte for both pins: 0x00

```
; Assume R1 points to the address of Port 9

mov r2, #0x00
strb r2, [r1, #0x0C]
```

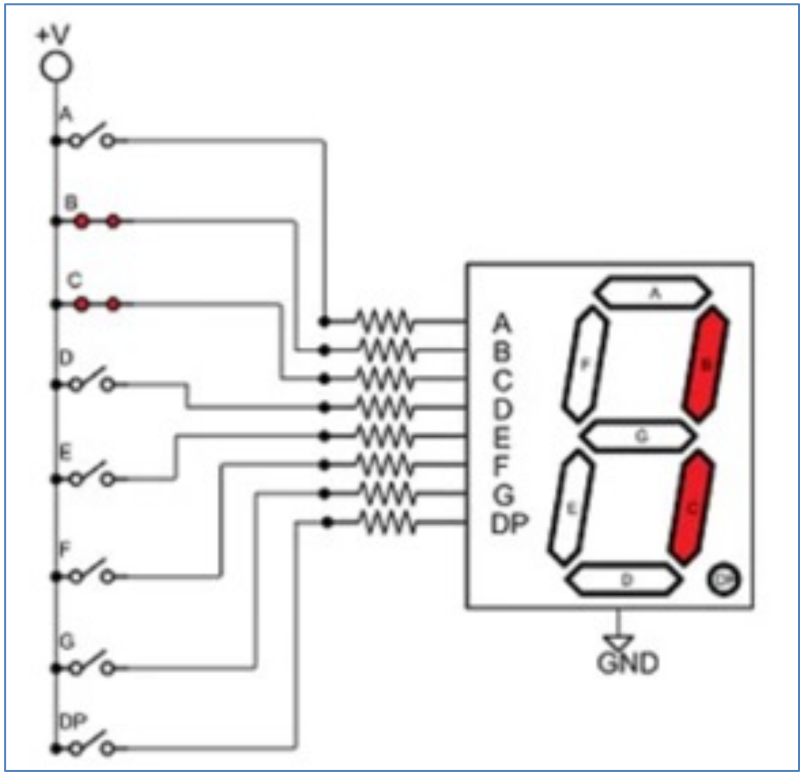
LEDs of 7-Segment Display

- Each segment is an individual LED
- Each LED is powered individually
- All cathodes are grounded
  - Common-Cathode system
- Note the order of letters, a, b, e, etc. to identify LEDs



LEDs of 7-Segment Display

- To display a particular number or character
    - The set of particular LEDs must be powered on by Microcontroller program
- Figure shows LEDs ‘b’ and ‘c’ powered ON to display ‘1’



7-Segment LED Interfacing

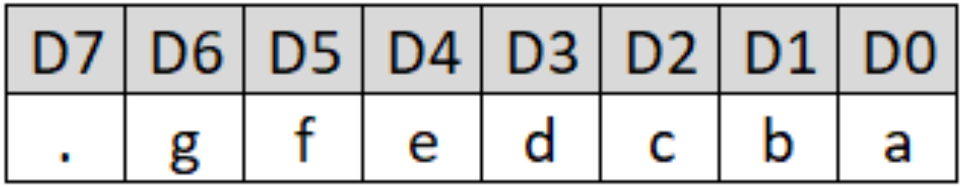
7-Segment LED can be

- Common Anode
  - All LEDs are connected to positive supply voltage
  - Microcontroller has to drive individual cathode LOW for current to flow to make LEDs light up
- Common Cathode
  - All cathodes of LEDs are connected to ground
  - Microcontroller pins must provide sufficient source current for each LED segment

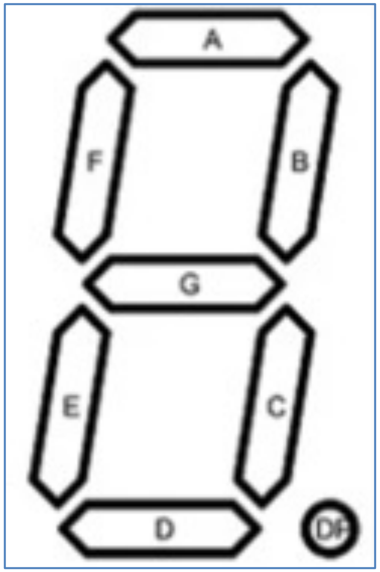
- An 8-bit port is sufficient to drive all segments (Port 4)

Pin Assignment

- Figure shows a desirable sequence of Pin assignment

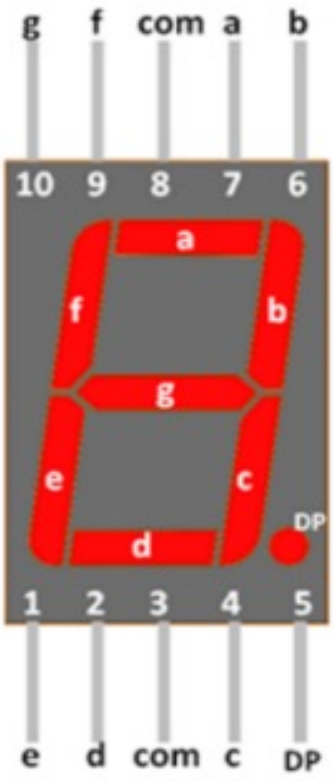


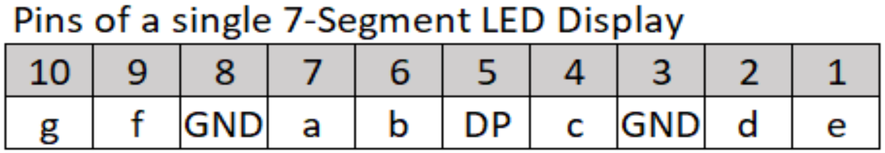
D7	D6	D5	D4	D3	D2	D1	D0
.	g	f	e	d	c	b	a



Pinout of a Single 7-Segment Display

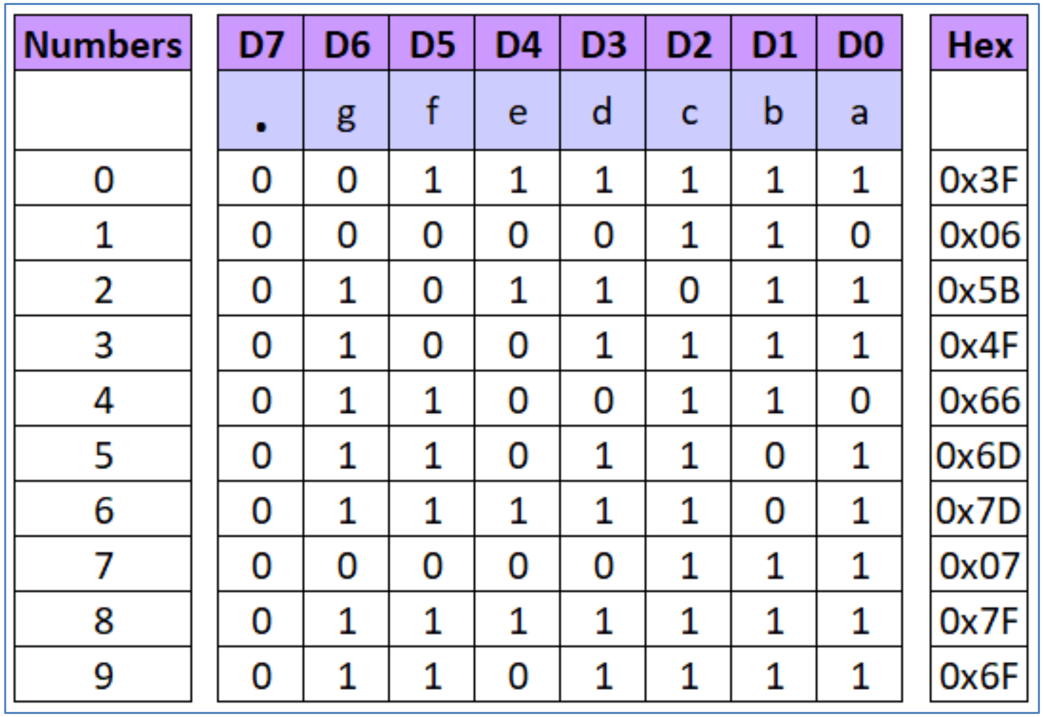
- Single 7-segment Display (Common Cathode – 3461BS)




Pins of a single 7-Segment LED Display

10	9	8	7	6	5	4	3	2	1
g	f	GND	a	b	DP	c	GND	d	e

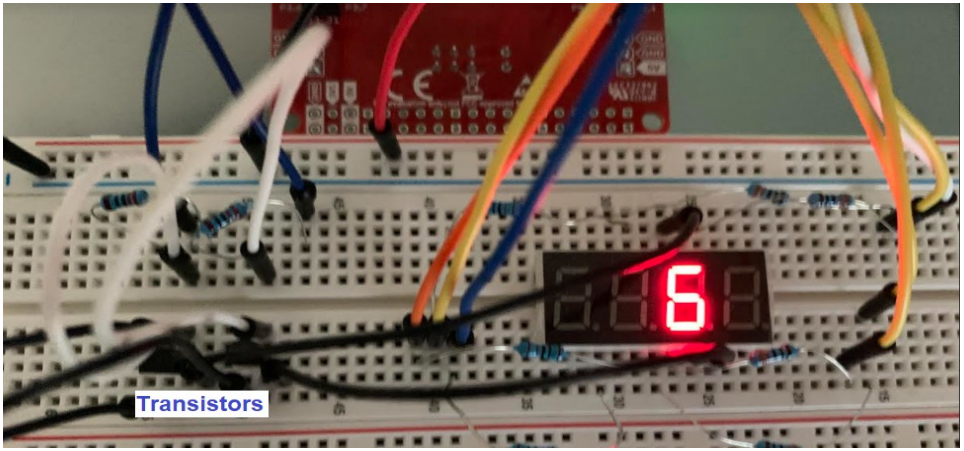
DP - Decimal Point

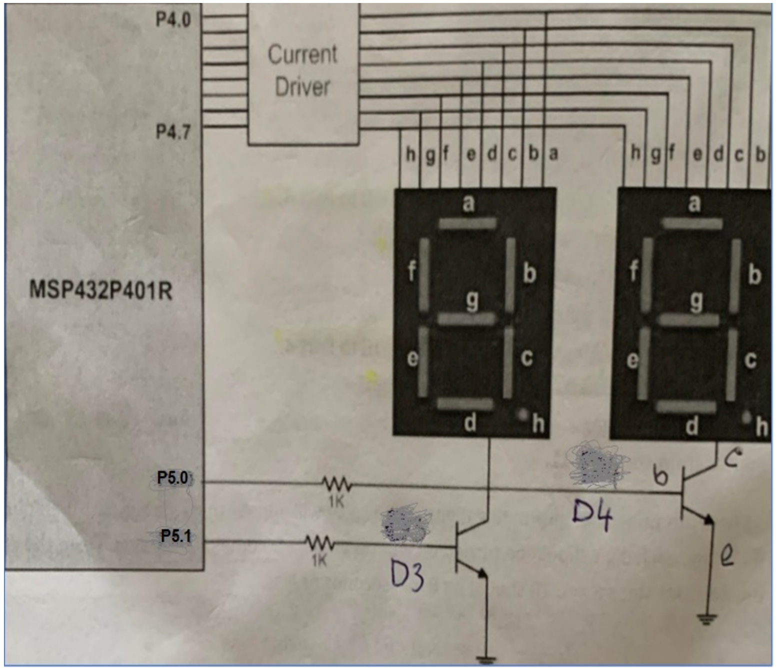


Numbers	D7	D6	D5	D4	D3	D2	D1	D0	Hex
	.	g	f	e	d	c	b	a	
0	0	0	1	1	1	1	1	1	0x3F
1	0	0	0	0	0	1	1	0	0x06
2	0	1	0	1	1	0	1	1	0x5B
3	0	1	0	0	1	1	1	1	0x4F
4	0	1	1	0	0	1	1	0	0x66
5	0	1	1	0	1	1	0	1	0x6D
6	0	1	1	1	1	1	0	1	0x7D
7	0	0	0	0	0	1	1	1	0x07
8	0	1	1	1	1	1	1	1	0x7F
9	0	1	1	0	1	1	1	1	0x6F

Microcontroller Connections for a Single 7-segment Display

- Two 7-Segement LEDS for two digits




P4.0
Current
Driver
P4.7
MSP432P401R
h g f e d c b a
h g f e d c b
a
f
b
g
e
c
d
h
a
f
b
g
e
c
d
h
P5.0
1K
P5.1
1K
D3
D4
b
c
e