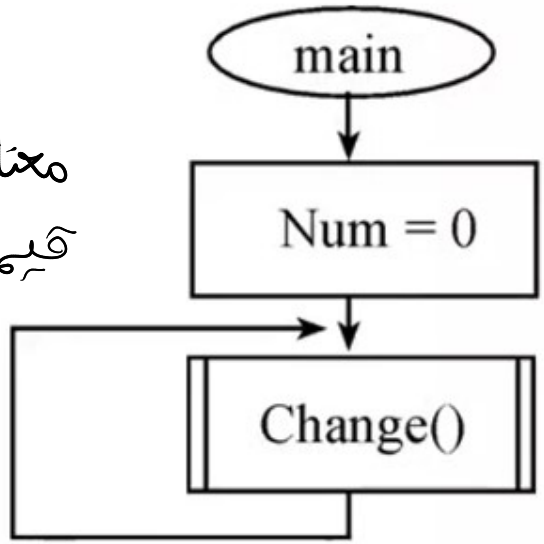
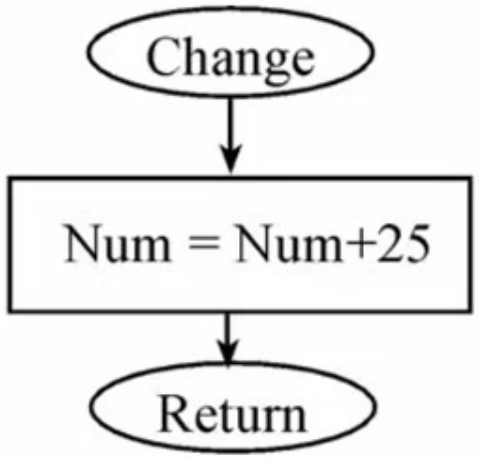
بسم الله الرحمن الرحيم

Functions

BX LR
معناه إنه يرجع
قيمة
return
address
LR جوا ال
↳ Link
Register






```
Change  LDR   R1,=Num    ; 5) R1 = &Num
        LDR   R0,[R1]    ; 6) R0 = Num
        ADD   R0,R0,#25  ; 7) R0 = Num+25
        STR   R0,[R1]    ; 8) Num = Num+25
        BX    LR         ; 9) return
main    LDR   R1,=Num    ; 1) R1 = &Num
        MOV   R0,#0      ; 2) R0 = 0
        STR   R0,[R1]    ; 3) Num = 0
loop    BL    Change     ; 4) function call
        B     loop       ; 10) repeat
```

```
unsigned long Num;
void Change(void){
  Num = Num+25;
}
void main(void){
  Num = 0;
  while(1){
    Change();
  }
}
```

Spale → بتحجز مكان
uninitialized Variables

Array Example

```
for(int i=0; i< 5;i++)
  {
    aa[i] = i;
    bb[i] = 5;
  }
```

عندي 2-array وهخزن فيهم ←

يعني أنا محتاج أفتح مكان في الميموري

```
        AREA    DATA
aa      SPACE 40
bb      SPACE 40

main    MOV R4,#0  ; i=0
        MOV R3,#5
forloop2
        CMP R4,#5  ; is i<5
        BGE forDone2
        LDR R0,=aa  ;aa[i] = aa+4*i
        ASL R2,R4,#2 ; R2=i*4
        STR R4,[R0,R2]
        LDR R6,=bb
        STR R3,[R6,R2] ; bb+i*4
        ADD R4,R4,#1
        B  forloop2
forDone2
```

ينفع أستخدم LSL عادي الدكتور قال كده في المحاضرة ← ASL

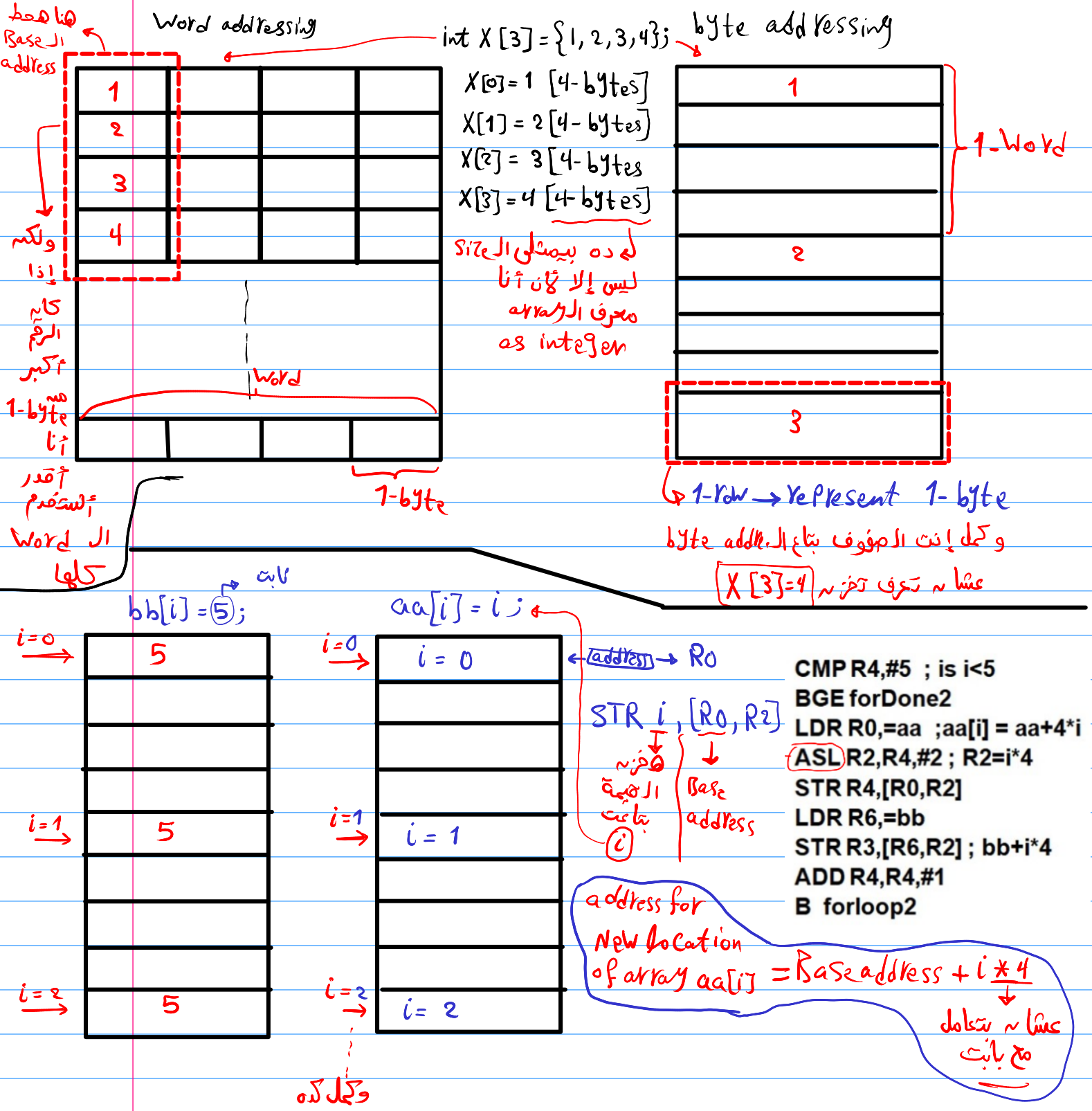
ASL R2, R4, #2

R4 ← ده ال Counter بتاع

$R_2 = i*4$

R_2 → temp register

i*4 → عشان أحول ال byte ل Word ضربت (4*)



GPIO → exactly like GPR يعني تقدر تستخدم بزيادة

Special i/o Ports → like Communication Protocall (UART, ——)
We use PA0, PA1 for that

I/O pins on Cortex-M microcontrollers have a wide range of alternative functions:

- **UART** — **Universal asynchronous receiver/transmitter**
- **SSI** — **Synchronous serial interface**
- **I^2C** — **Inter-integrated circuit**
- **Timer** — **Periodic interrupts, input capture, and output compare**
- **PWM** — **Pulse width modulation**
- **ADC** — **Analog to digital converter, measure analog signals**
- **Analog Comparator** — **Compare two analog signals**
- **QEI** — **Quadrature encoder interface**
- **USB** — **Universal serial bus**
- **Ethernet** — **High-speed network**
- **CAN** — **Controller area network**

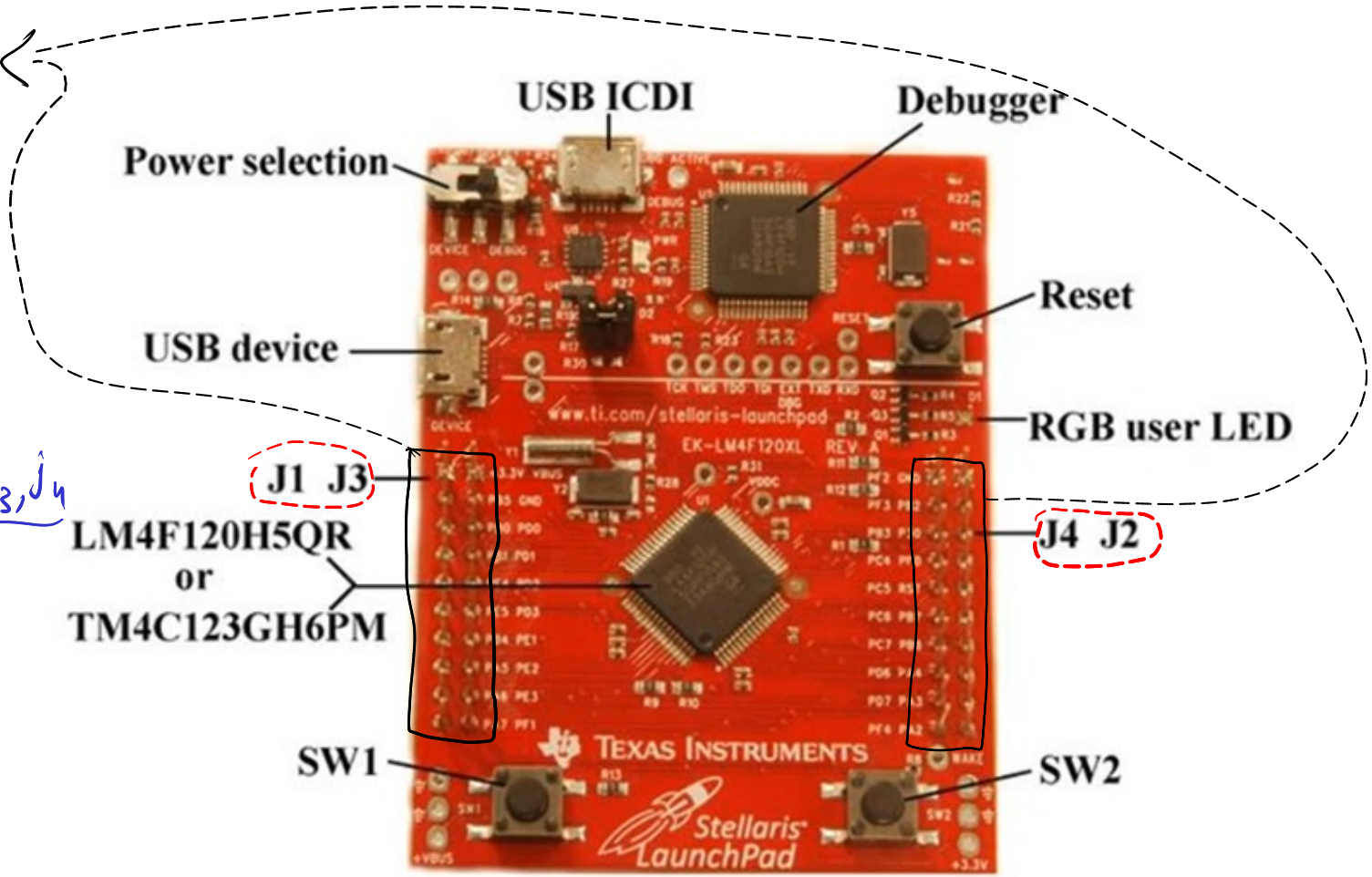
The UART can be used for serial communication between computers. It is asynchronous and allows for simultaneous communication in both directions. The SSI is alternately called serial peripheral interface (SPI). It is used to interface medium-speed I/O devices. In this book, we will use it to interface a graphics display.

We could use SSI to interface a digital to analog converter (DAC) or a secure digital card (SDC). I2C is a simple I/O bus that we will use to interface low speed peripheral devices. Input capture and output compare will be used to create periodic interrupts and measure period, pulse width, phase, and frequency. PWM outputs will be used to apply variable power to motor interfaces. In a typical motor controller, input capture measures rotational speed, and PWM controls power. A PWM output can also be used to create a DAC. The ADC will be used to measure the amplitude of analog signals and will be important in data acquisition systems. The analog comparator takes two analog inputs and produces a digital output depending on which analog input is greater. The QEI can be used to interface a brushless DC motor.

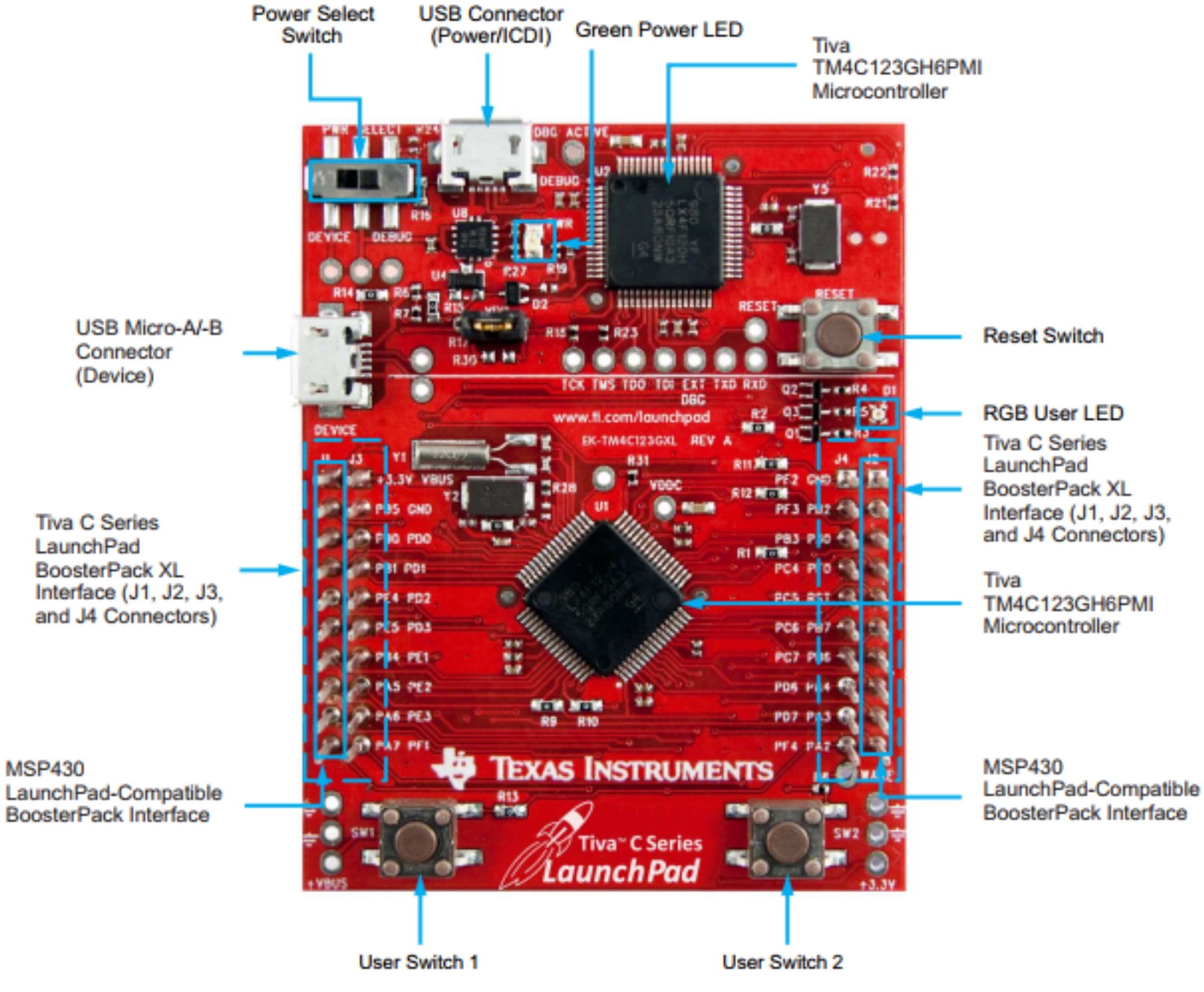
USB is a high-speed serial communication channel. The Ethernet port can be used to bridge the microcontroller to the Internet or a local area network. The CAN creates a high-speedcommunication channel between microcontrollers and is commonly found in automotive and other distributed control applications.

Tiva C Board

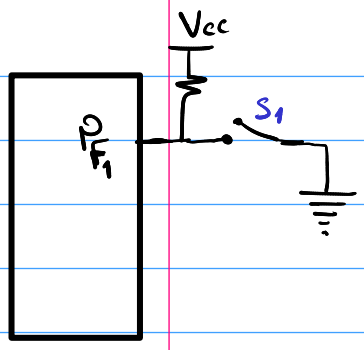
i/o Ports



*for J_1, J_2, J_3, J_4


Power Select Switch
USB Connector (Power/ICDI)
Green Power LED
Tiva TM4C123GH6PMI Microcontroller
USB Micro-A/-B Connector (Device)
Reset Switch
RGB User LED
Tiva C Series LaunchPad BoosterPack XL Interface (J1, J2, J3, and J4 Connectors)
Tiva C Series LaunchPad BoosterPack XL Interface (J1, J2, J3, and J4 Connectors)
Tiva TM4C123GH6PMI Microcontroller
MSP430 LaunchPad-Compatible BoosterPack Interface
MSP430 LaunchPad-Compatible BoosterPack Interface
User Switch 1
User Switch 2
TEXAS INSTRUMENTS
Tiva™ C Series LaunchPad
www.ti.com/launchpad
EK-TM4C123GXL REV A

Pull-up

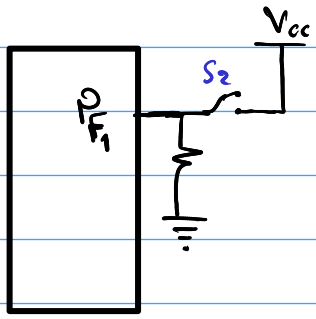


P_{F_1} → ~~forbled~~ floated

S_1 is "off" → P_{F_1} → high

S_1 is "on" → P_{F_1} → 0

Pull-down



S_2 is "off" → P_{F_1} → low

S_2 is "on" → P_{F_1} → high

* Pull up & Pull down Concept is used for switching

Recall D-flip/flop

D Flip-flop

Symbol

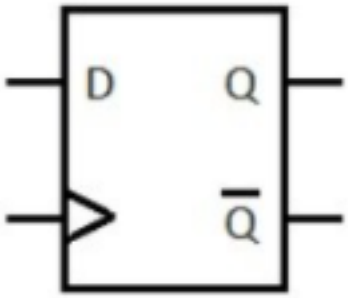


Table of truth:

clk	D	Q	$\overline{Q}$
0	0	Q	$\overline{Q}$
0	1	Q	$\overline{Q}$
1	0	0	1
1	1	1	0

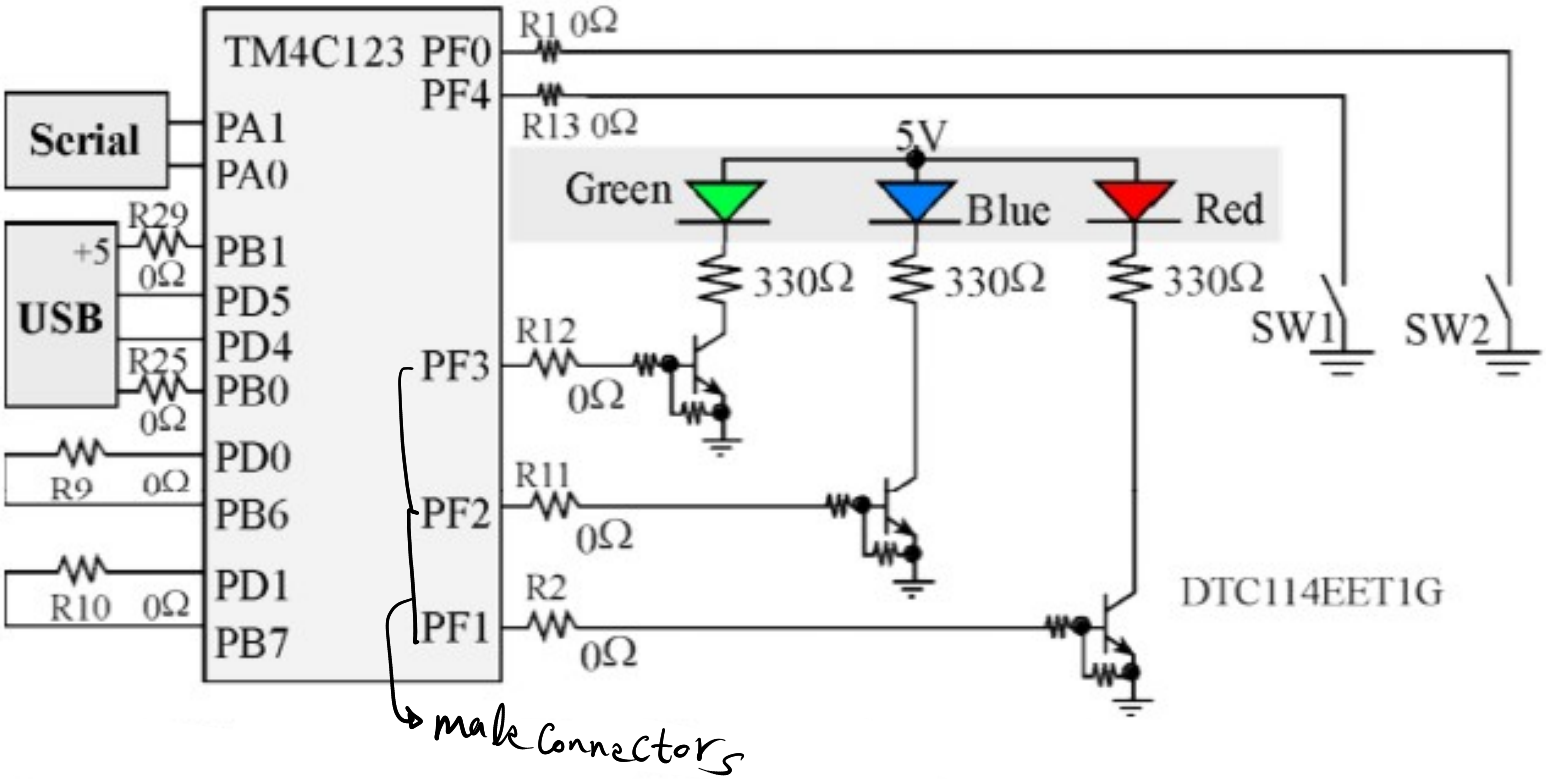
Q: not change if clk low

Q = D

"output" "input"

if clk → high

LaunchPad Switches and LEDs



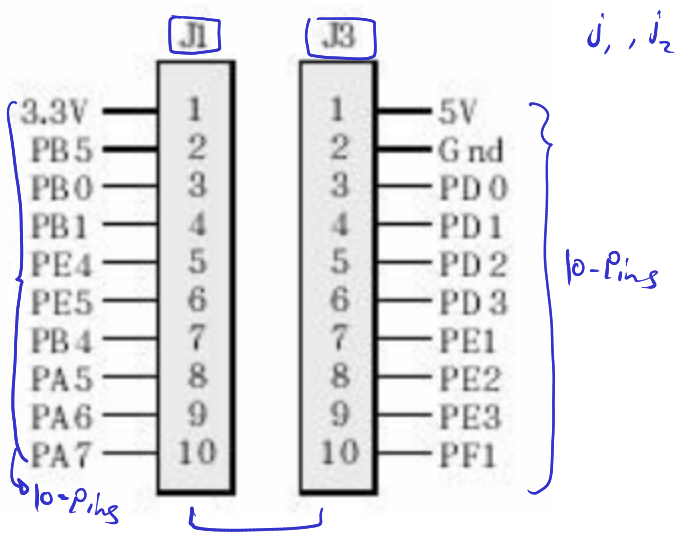
Imp.

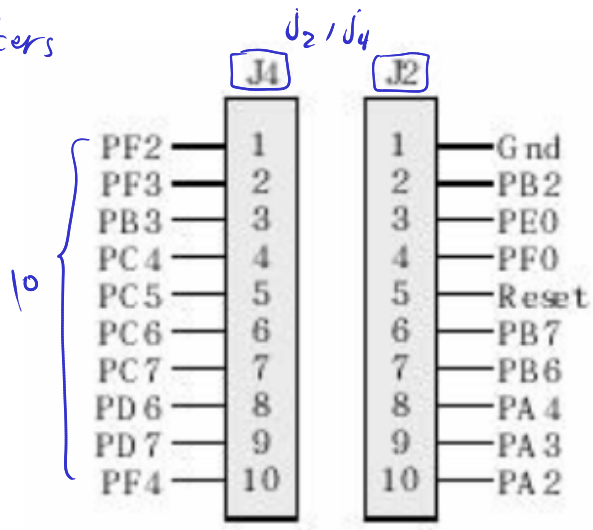
Pins PA1 – PA0 create a serial port, which is linked through the debugger cable to the PC. The serial link is a physical UART as seen by the LM4F/TM4C and mapped to a virtual COM port on the PC. The USB device interface uses PD4 and PD5. The JTAG debugger requires pins PC3 – PC0. The LaunchPad connects PB6 to PD0, and PB7 to PD1. If you wish to use both PB6 and PD0 you will need to remove the R9 resistor. Similarly, to use both PB7 and PD1 remove the R10 resistor.

The Tiva® LaunchPad evaluation board has two switches and one 3-color LED. See Figure 4.6. The switches are negative logic and will require activation of the internal pull-up resistors. In particular, you will set bits 0 and 4in **GPIO_PORTF_PUR_R** register. The LED interfaces on PF3 – PF1 are positive logic. To use the LED, make the PF3 – PF1 pins an output. To activate the red color, output a one to PF1. The blue color is on PF2, and the green color is controlled by PF3. The 0-Ω resistors (R1, R2, R11, R12, R13, R25, and R29) can be removed to disconnect the corresponding pin from the external hardware.

The LaunchPad has four 10-pin connectors, labeled as J1 J2 J3 J4 in Figures 4.5 and 4.7, to which you can attach your external signals. The top side of these connectors has male pins, and the bottom side has female sockets. The intent is to stack boards together to make a layered system, see Figure 4.7. Texas Instruments also supplies Booster Packs, which are pre-made external devices that will plug into this 40-pin connector. The Booster Packs for the MSP430 LaunchPad are compatible (one simply plugs these 20-pin connectors into the outer two rows) with this board. The inner 10-pin headers (connectors J3 and J4) are not intended to be compatible with other TI LaunchPads. J3 and J4 apply only to Stellaris/Tiva Booster Packs.

There are two methods to connect external circuits to the LaunchPad. One method uses male to female jumper cable (e.g., item number 826 at www.adafruit.com) or solder a solid wire into a female socket (e.g., Hirose DF11-2428SCA) creating a male to female jumper wire. The second method uses male-male wires and connect to the bottom of the LaunchPad.





imp.

Table 2-2. User Switches and RGB LED Signals

GPIO Pin	Pin Function	USB Device
PF4	GPIO	SW1
PF0	GPIO	SW2
PF1	GPIO	RGB LED (Red)
PF2	GPIO	RGB LED (Blue)
PF3	GPIO	RGD LED (Green)

General Purpose i/o Ports

Registers

Table 2-7. In-Circuit Debug Interface (ICDI) Signals

GPIO Pin	Pin Function
PC0	TCK/SWCLK
PC1	TMS/SWDIO
PC2	TDI
PC3	TDO/SWO

Table 2-8. Virtual COM Port Signals

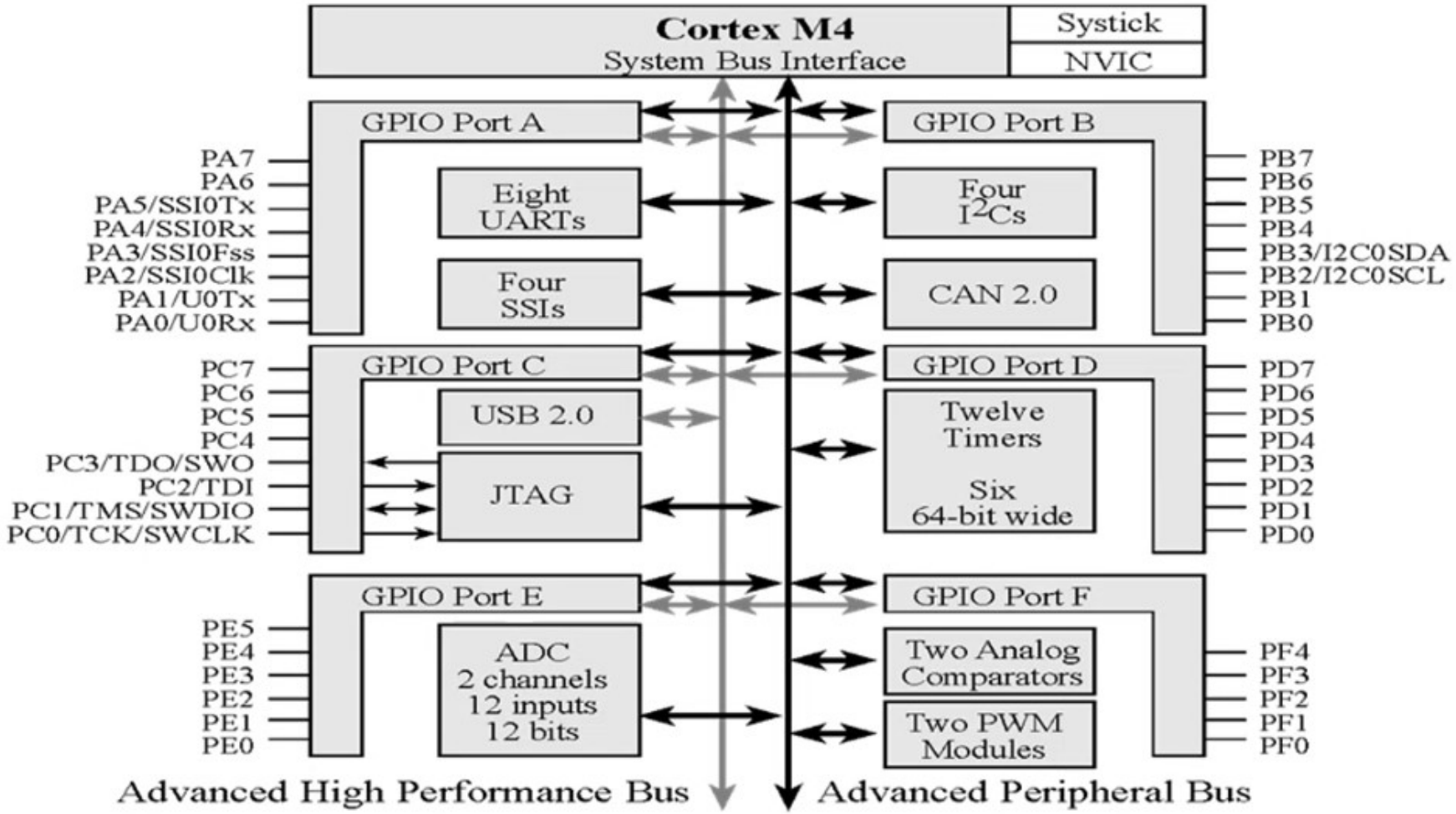
GPIO Pin	Pin Function
PA0	U0RX
PA1	U0TX

Special i/o Ports

Table 2-1. USB Device Signals

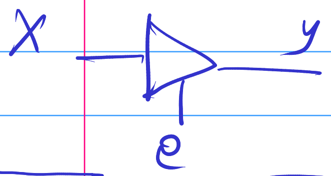
GPIO Pin	Pin Function	USB Device
PD4	USB0DM	D–
PD5	USB0DP	D+

Texas Instruments TM4C123



recall:

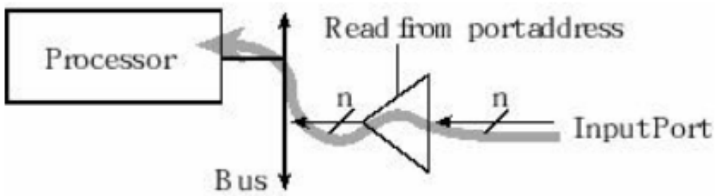
tristate buffer

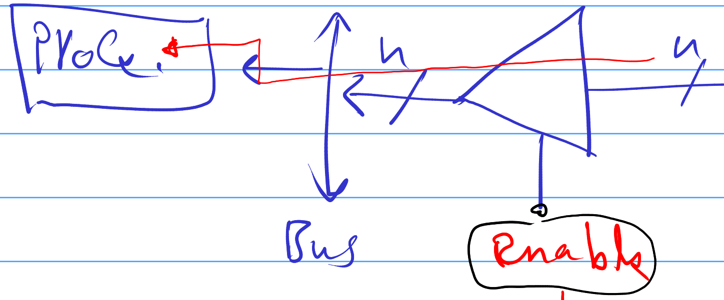


e=1 | e=0

X=y | high impedance X ≠ y

The simplest I/O port on a microcontroller is the parallel port. A parallel I/O port is a simple mechanism that allows the software to interact with external devices. It is called parallel because multiple signals can be accessed all at once. An **input port**, which is read only, allows the software to read external digital signals. That means a read cycle access from the port address returns the values existing on the inputs at that time. In particular, the tristate driver (triangle shaped circuit in Figure 4.11) will drive the input signals onto the data bus during a read cycle from the port address. A write cycle access to an input port usually produces no effect. The digital values existing on the input pins are copied into the microcontroller when the software executes a read from the port address. There are no digital input-only ports on the TM4C family of microcontrollers. TM4C microcontrollers have 5V-tolerant digital inputs, meaning an input voltage from 2.0 to 5.0 V will be considered high, and a voltage from 0 to 1.3 V will be considered as low. Check to see the pins on your microcontroller are 5-V tolerant.





i/Port → use to connect software with micro controller [USB]

→ Read from port address = 1
"off" = 0

* لو أنا مقفل الـ read port ساعتها حتى ولو كتبت مش هيأثر عليها ~ بمجرد ما أفتح الـ read port حصلها الـ execute القيمة على طول

* بالنسبة لـ input في قيمة تسمى tolerance أحدد إن low والـ high

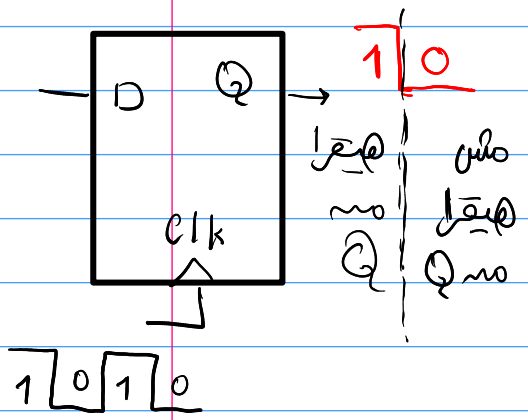
from 2 → 5V [high]

〃 0 → 1.3V [low]

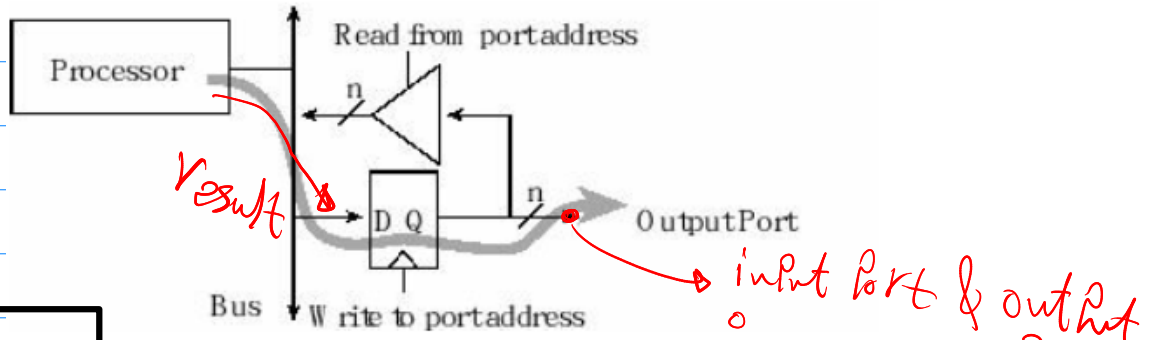
What happens if the software writes to an input port ?

Ans: Nothing will happen.

recall:



While an input device usually just involves the software reading the port, an output port can participate in both the read and write cycles very much like a regular memory. Figure 4.12 describes a **readable output port**. A write cycle to the port address will affect the values on the output pins. In particular, the microcontroller places information on the data bus and that information is clocked into the D flip-flops. Since it is a readable output, a read cycle access from the port address returns the current values existing on the port pins. There are no output-only ports on the TM4C family of microcontrollers.



input port & output port

* لو أنا كتبت على ال port i/o هيحصل تغير على ال output

وكنا قولنا أنا بنفع أكتب وأقرا في حالة input

What happens if the software reads from an output port ?

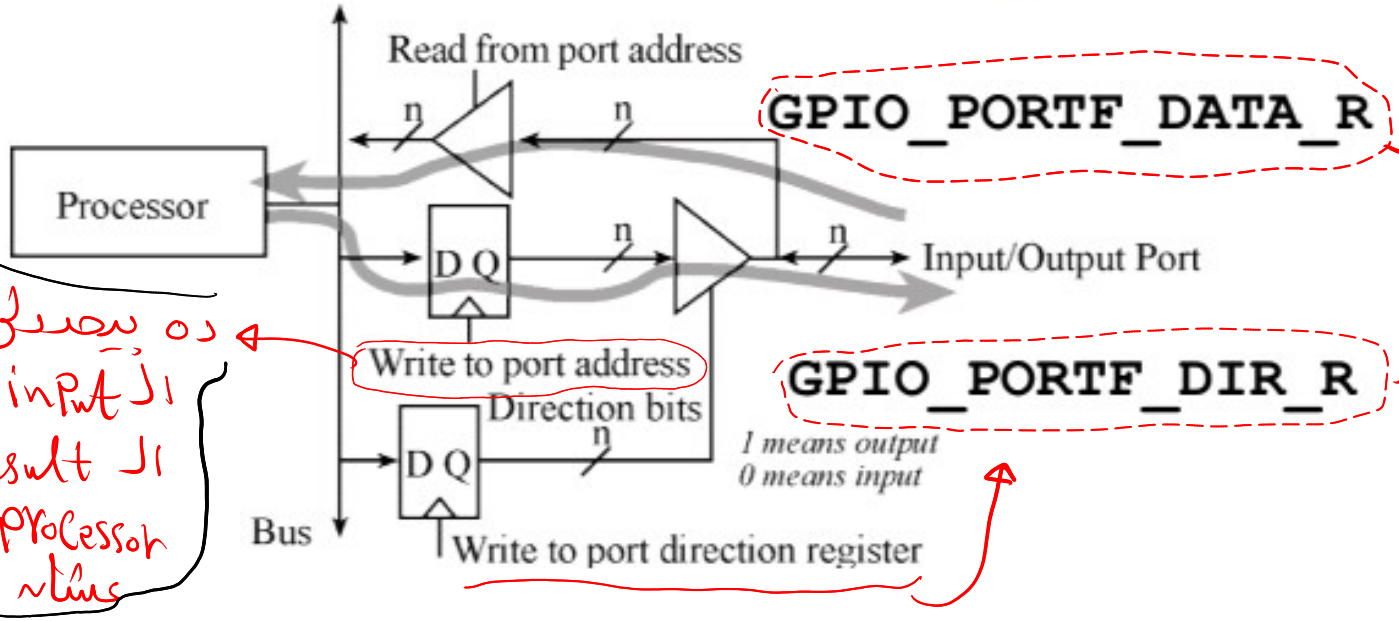
Ans: If the software reads this output port it gets the values last written to the port. For example, if the user mistakenly grounded the output pin (very bad thing to do), and the software writes a '1'; when it reads it will get '1'

* دلوقتي أنا عاوز أتحكم بالحالة بتاعتي أنا والـ input والـ output يعني راجع للـ Process ولا جاي من الـ Process

↓

direction

I/O Ports and Control Registers



GPIO_PORTF_DATA_R

GPIO_PORTF_DIR_R

ده يتحدد أنا هكتب على الـ input وبالتالي مبعتش الـ result اللي جاية من الـ Processor متروحش على الـ Port عشان أنا هكتب عليها وبالتالي مش هتغير في الـ output

ده خاص بالـ Data اللي هترجع أو هتيجي من الـ Proc

The input/output direction of a bidirectional port is specified by its direction register.

GPIO_PORTF_DIR_R, specify if corresponding pin is input or output:

- 0 means input
- 1 means output

DEN → Digital enable register

I/O Programming

Address	7	6	5	4	3	2	1	0	Name
$400F.E108	--	--	GPIOF	GPIOE	GPIOD	GPIOC	GPIOB	GPIOA	SYSCTL_RCGC2_R
$4000.43FC	DATA	DATA	DATA	DATA	DATA	DATA	DATA	DATA	GPIO_PORTA_DATA_R
$4000.4400	DIR	DIR	DIR	DIR	DIR	DIR	DIR	DIR	GPIO_PORTA_DIR_R
$4000.4420	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	GPIO_PORTA_AFSEL_R
$4000.4510	PUE	PUE	PUE	PUE	PUE	PUE	PUE	PUE	GPIO_PORTA_PUR_R
$4000.451C	DEN	DEN	DEN	DEN	DEN	DEN	DEN	DEN	GPIO_PORTA_DEN_R
$4000.4524	1	1	1	1	1	1	1	1	GPIO_PORTA_CR_R
$4000.4528	0	0	0	0	0	0	0	0	GPIO_PORTA_AMSEL_R
$4000.53FC	DATA	DATA	DATA	DATA	DATA	DATA	DATA	DATA	GPIO_PORTB_DATA_R
$4000.5400	DIR	DIR	DIR	DIR	DIR	DIR	DIR	DIR	GPIO_PORTB_DIR_R
$4000.5420	SEL	SEL	SEL	SEL	SEL	SEL	SEL	SEL	GPIO_PORTB_AFSEL_R
$4000.5510	PUE	PUE	PUE	PUE	PUE	PUE	PUE	PUE	GPIO_PORTB_PUR_R
$4000.551C	DEN	DEN	DEN	DEN	DEN	DEN	DEN	DEN	GPIO_PORTB_DEN_R
$4000.5524	1	1	1	1	1	1	1	1	GPIO_PORTB_CR_R
$4000.5528	0	0	AMSEL	AMSEL	0	0	0	0	GPIO_PORTB_AMSEL_R
$4000.63FC	DATA	DATA	DATA	DATA	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_DATA_R
$4000.6400	DIR	DIR	DIR	DIR	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_DIR_R
$4000.6420	SEL	SEL	SEL	SEL	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_AFSEL_R
$4000.6510	PUE	PUE	PUE	PUE	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_PUR_R
$4000.651C	DEN	DEN	DEN	DEN	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_DEN_R
$4000.6524	1	1	1	1	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_CR_R
$4000.6528	AMSEL	AMSEL	AMSEL	AMSEL	JTAG	JTAG	JTAG	JTAG	GPIO_PORTC_AMSEL_R
$4000.73FC	DATA	DATA	DATA	DATA	DATA	DATA	DATA	DATA	GPIO_PORTD_DATA_R
$4000.7400	DIR	DIR	DIR	DIR	DIR	DIR	DIR	DIR	GPIO_PORTD_DIR_R

On most embedded microcontrollers, the I/O ports are memory mapped. This means the software can access an input/output port simply by reading from or writing to the appropriate address. It is important to realize that even though I/O operations "look" like reads and writes to memory variables, the I/O ports often DO NOT act like memory. For example, some bits are read-only, some are write-only, some can only be cleared, others can only be set, and some bits cannot be modified. To make our software easier to understand we include symbolic definitions for the I/O ports. We set the direction register(e.g., **GPIO_PORTF_DIR_R**) to specify which pins are input and which are output. Individual port pins can be general purpose I/O (GPIO) or have an alternate function. We will set bits in the alternate function register (e.g., **GPIO_PORTF_AFSEL_R**) when we wish to activate the alternate functions listed in Tables 4.1, 4.3, and 4.4. To use a pin as a digital input or output, we must set the corresponding bit in the digital enable register(e.g., **GPIO_PORTF_DEN_R**). To use a pin as an analog input we must set the corresponding bit in theanalog mode select register (e.g., **GPIO_PORTF_AMSEL_R**). Typically, we write to the direction and alternate function registers once during the initialization phase. We use the data register(e.g., **GPIO_PORTF_DATA_R**) to perform the actual input/output on the port. Table 4.5 shows some of the parallel port registers for the TM4C123. Each of the ports has a clock, which can be separately enabled by writing to the **SYSCTL_RCGCGPIO_R** register.

Set Port Direction & Port Type

Pointed to Direction register

```
LDR     R1,= GPIO_PORTF_DIR_R
MOV     R0,#0x0E
STR     R0,[R1]

LDR     R1,=GPIO_PORTF_DEN_R
MOV     R0,#0xFF
STR     R0,[R1]
```

Processor ⇄ i/o Port

Good link

https://users.ece.utexas.edu/~valvano/Volume1/E-Book/C6_MicrocontrollerPorts.htm