Câu chuyện bắt đầu từ một cậu bé,
và một ý tưởng
có thể
làm thay đổi thế giới... PAY IT FORWARD

Đó là khi bạn giúp đỡ 3 người bạn không quen biết,
dù là bằng thời gian,
hay công sức,
hay kinh nghiệm,
hay kiến thức,
hay tiền bạc, ...
của mình.

Mà không chờ đợi một sự báo ân nào.

Chỉ cần mỗi người trong 3 người đó,
lại đem những gì mình có, mà người khác cần,
tiếp tục giúp đỡ thêm 3 người nữa.

Chính những người-giúp-đỡ, và người-được-giúp-đỡ,
sẽ là những người góp phần thay đổi thế giới...

Một thế giới sẻ chia kiến thức - và yêu thương ...

PAY IT FORWARD ...

Chúng tôi không sáng tạo ra câu nói này.

Pay it forward...

Hãy tri ân người giúp mình bằng cách giúp đỡ người khác
Cho đi không phải để nhận lại.

CÂU LẠC BỘ NGHIÊN CỨU KHOA HỌC KHOA ĐIỆN-ĐIỆN TỬ
ĐH BÁCH KHOA TP. HỒ CHÍ MINH

payitforward.edu.vn

MSP430 COURSE

LESSON 1

MCU & GPIO

Training document for C8 course – update 03-04-2013

Contents

Part 1:
MCUs & MSP430G2553

1.1 Why MCU (uC or µC)?

Requirement:

a. Blink a LED every 1s

b. Turn on a led when button is pushed

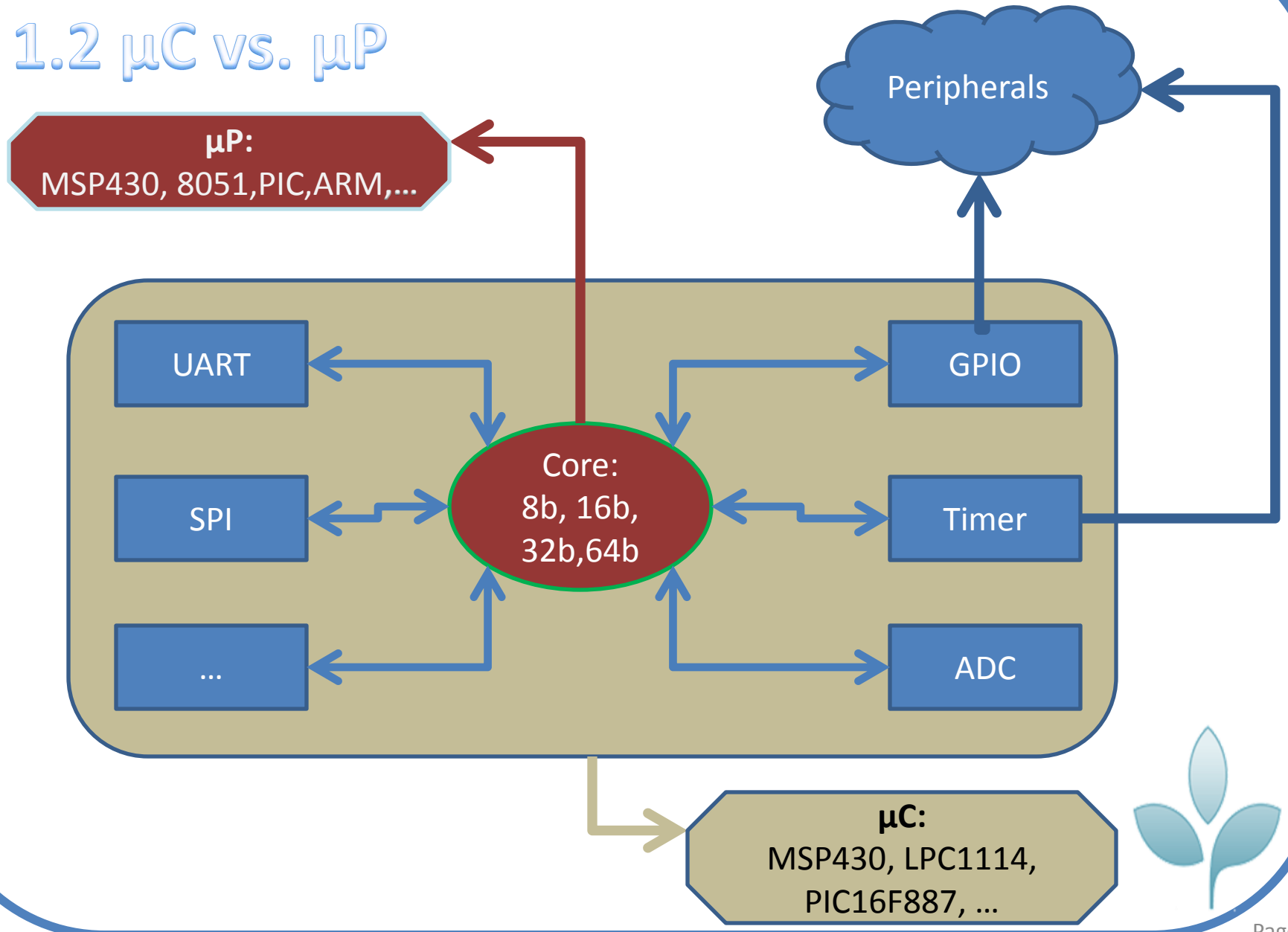
c. Timer, ADC, UART, Interrupt, SPI, I2C…

d. ….

Solution: Micro Controller Unit

- *Integrated circuits*
- *Programmable*

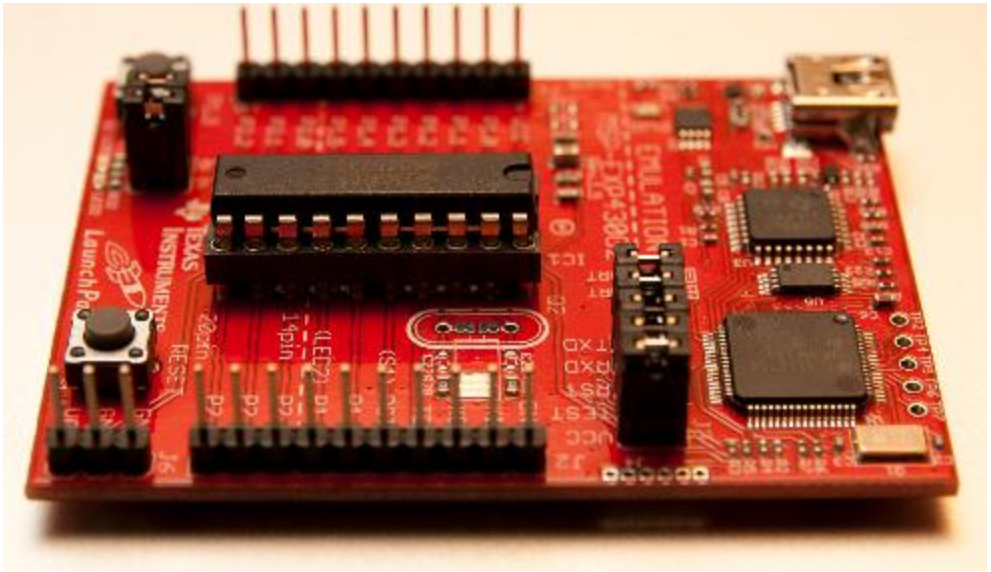
1.2 µC vs. µP


µP:
MSP430, 8051,PIC,ARM,…
Peripherals
UART
SPI
…
Core:
8b, 16b,
32b,64b
GPIO
Timer
ADC
µC:
MSP430, LPC1114,
PIC16F887, …

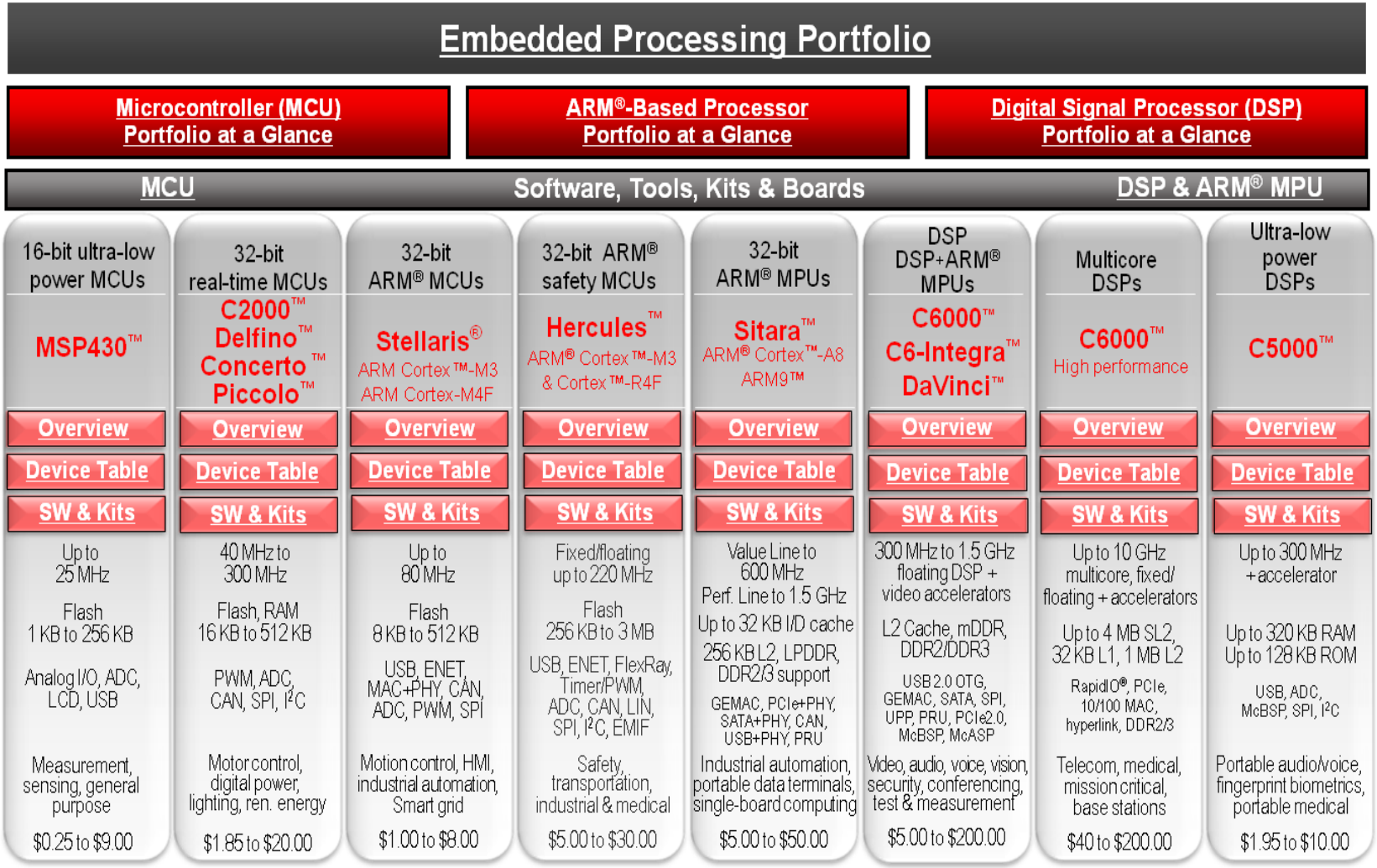
1.3 Why MSP430G2553?

- ***Simple:*** *hardware & software*
- ***Enough:*** price , *peripherals,...*
- ***Developing*** <-> ***TI Vietnam***

- 16kB Flash
- 512B RAM
- 2 Timer_A3's
- 8 Ch. Comp_A+
- 8 Ch. ADC10
- USCI

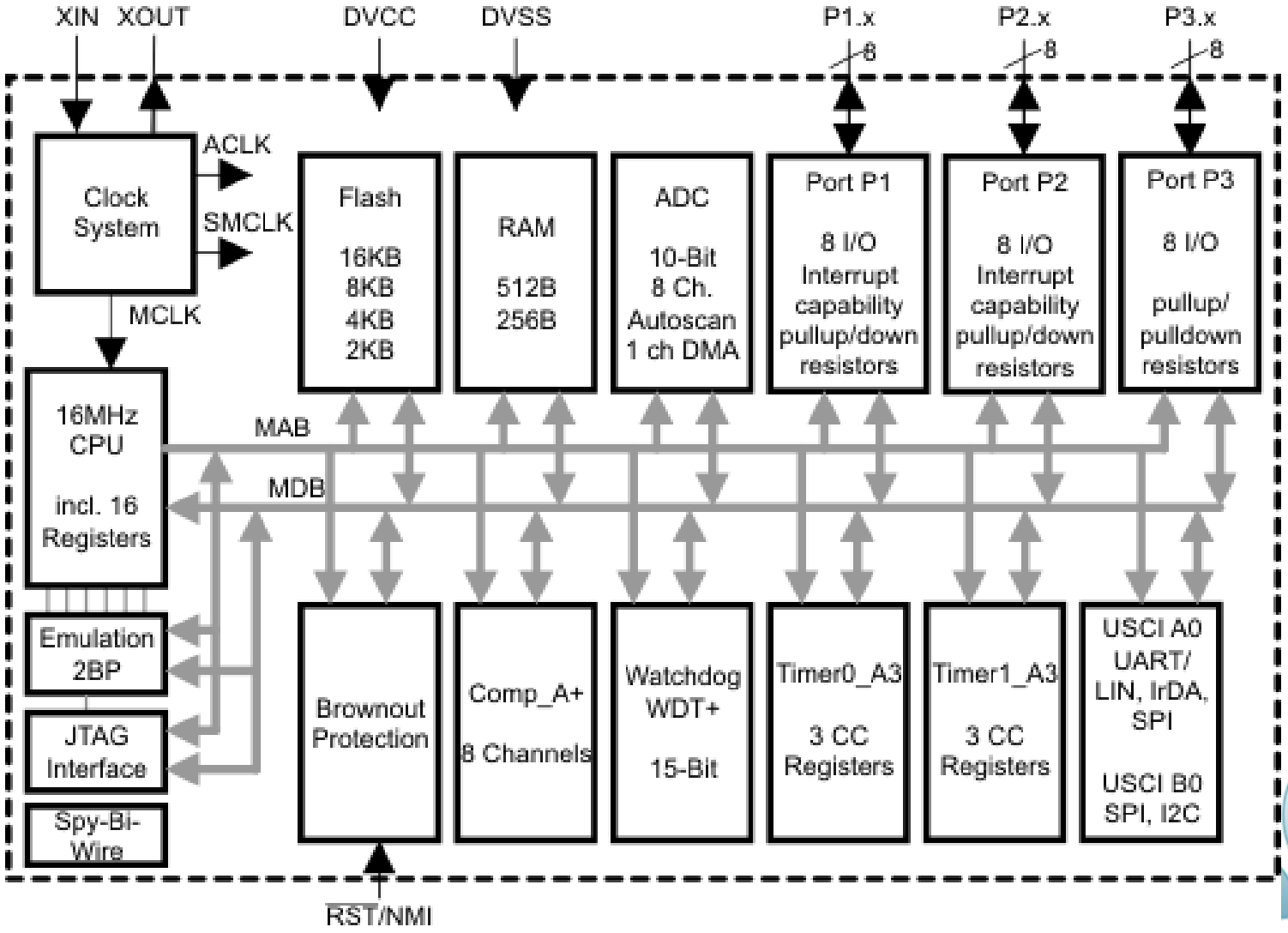


1.4 Overview of TI MCUs



MPUs – Microprocessors

1.4 Overview of MSP430G2x53



http://www.ti.com/product/msp430g2553

1.4 Overview of MSP430G2553

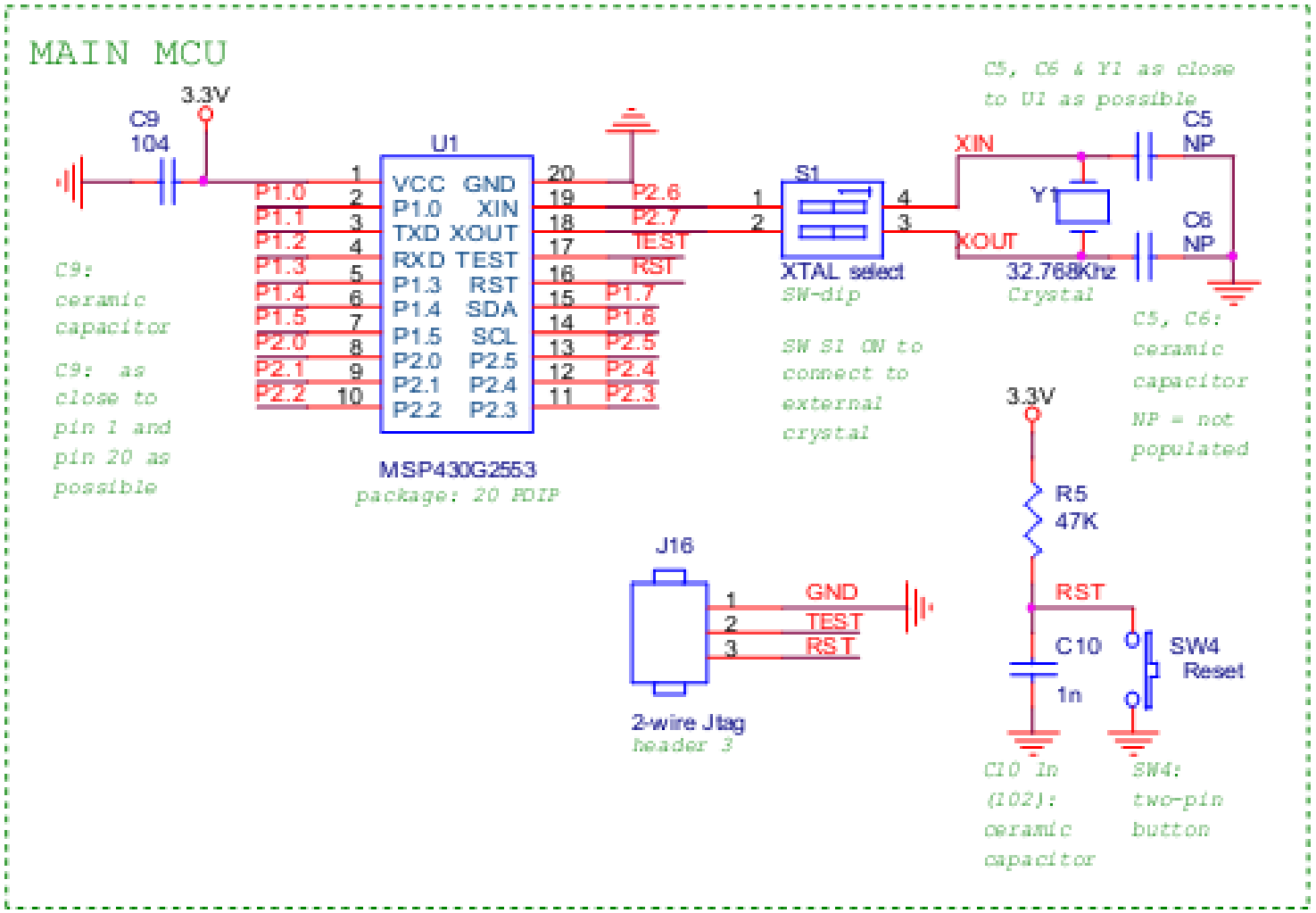
- Low Supply-Voltage Range
  - **1.8 V to 3.6 V**
- **Ultra-Low Power** Consumption
- **16-Bit RISC** Architecture, 62.5-ns Instruction Cycle Time
- Basic Clock Module Configurations
  - **Internal Frequencies up to 16 MHz** With Four Calibrated Frequency
  - **Internal Very-Low-Power Low-Frequency** (LF) Oscillator
  - 32-kHz **Crystal**
  - External **Digital Clock Source**

Package:

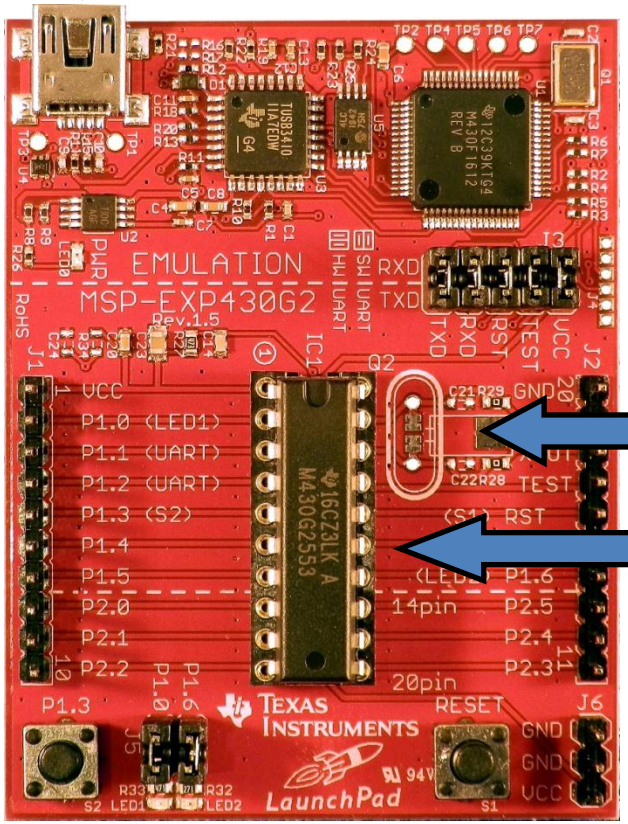
- TSSOP: 20 Pin, 28 Pin
- PDIP: 20 Pin
- QFN: 32 Pin

http://www.ti.com/product/msp430g2553

1.5 MSP430G2553 on board

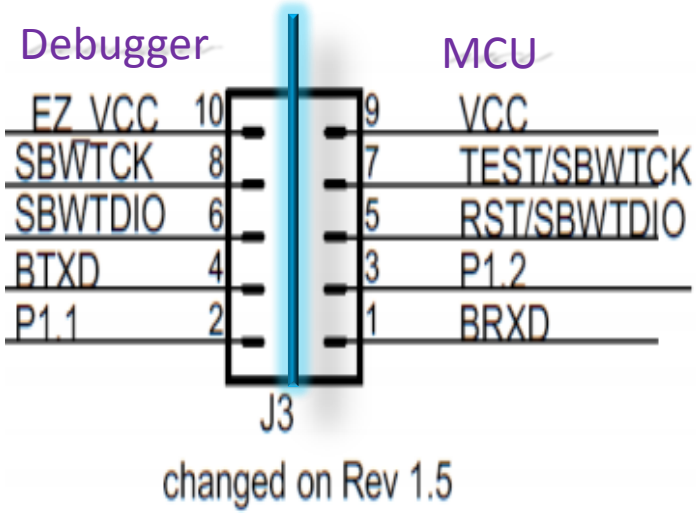


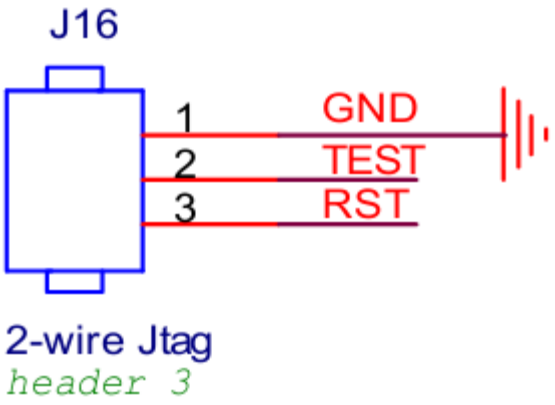
1.6 MSP430 Launch Pad

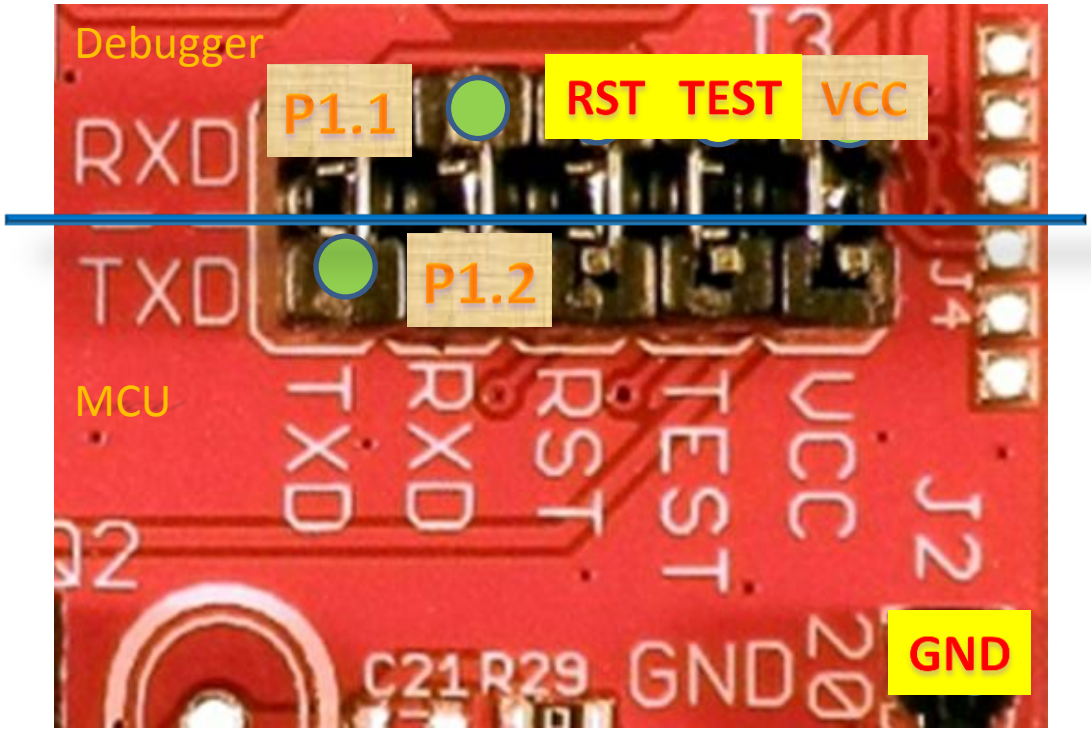


http://www.ti.com/tool/msp-exp430g2

1.6 Launch Pad -> Debugger

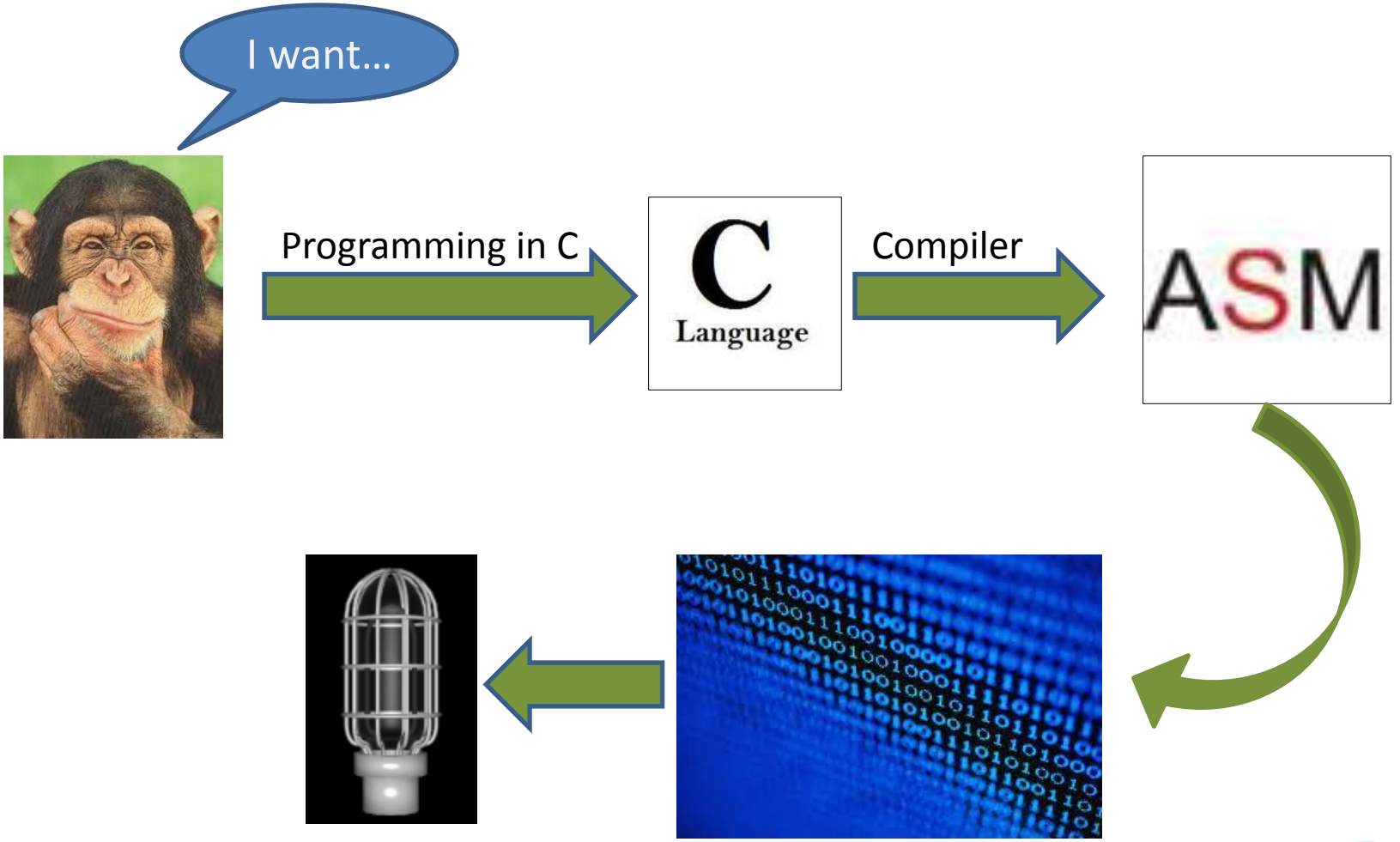






Part 2:
C programming
CCS IDE

2.1 C programming



2.1 C programming

```
49  ************************************
50
51  /************************************
52   * MAIN
53   ************************************
54
55  void main()
56  {
57      unsigned char c, i;
58      unsigned int val = 12345;
59      Config_stop_WDT();
60      Config_Clocks();
61      lcd_init ();
62      uart_init ();
63
64      __bis_SR_register(GIE);
65
66      while (1)
67      {
68          lcd_clear ();
```

```
52  }
53
54  //*************************************************
55  // Clocks Configurations
56  //*************************************************
57  void Config_Clocks(void)
58  {
59      if (CALBC1_1MHZ ==0xFF || CALDCO_1MHZ == 0xFF)
60      {
61        while(1);                       // If cal cons
62      }
63
64      BCSCTL1 = CALBC1_1MHZ;            // Set DCO ran
65      // BCSCTL1 - Basic Clock System Control Regist
66          // XT2OFF    = XT2 off. This bit turns off
67          // XTS       = LFXT1 mode select. 1: High-f
68          // DIVAx     = Divider for ACLK
69          // RSELx     = Range select
70      DCOCTL = CALDCO_1MHZ;             // Set DCO ste
71      // DCOCTL - DCO Control Register
72          // DCOx      = DCO frequency select
73          // MODx      = Modulator selection
```

2.3 IDE (Integrated Development Environment)

- 16KB Limit on Msp430 devices

Download:
http://processors.wiki.ti.com/index.php/Download_CCS

Tutor:

http://www.diendanti.com/showthread.php?113-IDE-Coding-v%E1%BB%9Bi-Code-Composer-Studio-5-%28CCS-5%29-S%E1%BB%AD-d%E1%BB%A5ng-cho-MSP430

IAR SYSTEMS
From Idea to Target®
www.iar.com

- 8KB Limit on MSP430X devices
- 16KB Limit on eZ430 devices

Download:
http://processors.wiki.ti.com/index.php/IAR_Embedded_Workbench_for_TI_MSP430

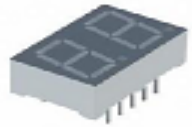
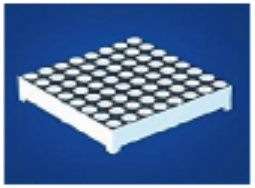
Tutor: http://www.diendanti.com/showthread.php?114-IDE-H%C6%B0%E1%BB%9Bng-d%E1%BA%ABn-s%E1%BB%AD-d%E1%BB%A5ng-IAR-Embedded-Workbench-cho-MSP430

2.3 IDE CCS (Code Composer Studio)

- Integrated development environment for **TI embedded processors**
  - Includes **debugger, compiler, editor, simulator, OS**…
  - The IDE is built on the Eclipse open source software framework
  - Extended by **TI** to support device capabilities
- Integrate additional tools
  - OS application development tools (Linux, Android…)
  - Code analysis, source control…
- Linux support
- Low cost! $445 or $495 (!)
  - -> Free code size limited for students

Part 3:
GPIOs IN MSP430

General Purpose Input/ Output

- **Learn**
  - CCS IDE
  - C language for MCU
  - Overview of MSP430G2553
  - Buttons, leds
- **To do**
  - 7-segment LED 
  - Keypad 4x4 
  - 8x8 Matrix LED 
  - LCD 16x2 

3.1 Registers: PxSEL (x:1 , 2) and PxSEL2(x:1 , 2)

Các Thanh ghi này qui định chế độ làm việc cho các chân bao gồm PxSEL và PxSEL2

PxSEL2	PxSEL	Pin Function
0	0	I/O function is selected.
0	1	Primary peripheral module function is selected.
1	0	Reserved. See device-specific data sheet.
1	1	Secondary peripheral module function is selected.

VD: Ta muốn port 1 là GPIO thì ta thực hiện lệnh như sau:

```
P1SEL  = 0;
P1SEL2 = 0;
```

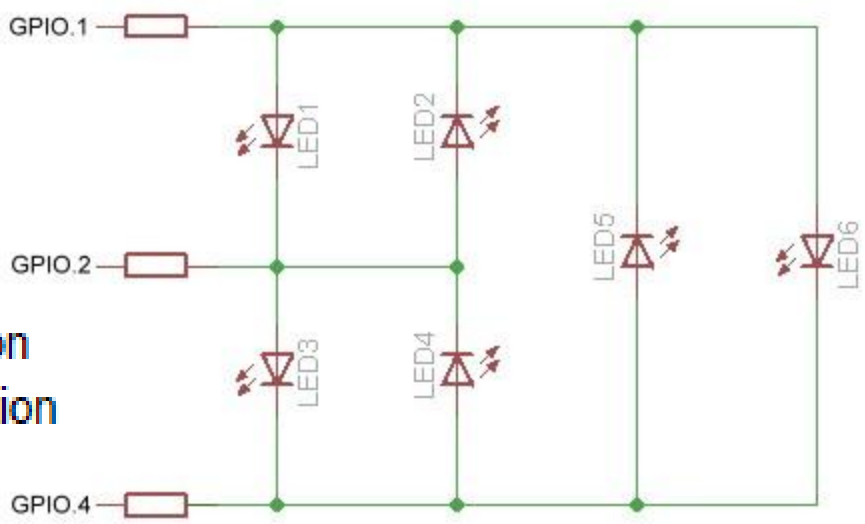
Lưu ý: Interrupts P1 và P2 sẽ bị vô hiệu hóa khi PxSEL=1

(!) Xem thêm Datasheet: Port Schematics

3.1 Registers: PxDIR (x:1 , 2)

Thanh ghi này qui định loại ngõ vào ra cho các chân GPIO

Mặc định tất cả các chân là input (0), khi cần đổi thành output (1), ta đặt Bit tương ứng trên thanh ghi là 1.



Bit = 0: The port pin is switched to input direction
Bit = 1: The port pin is switched to output direction

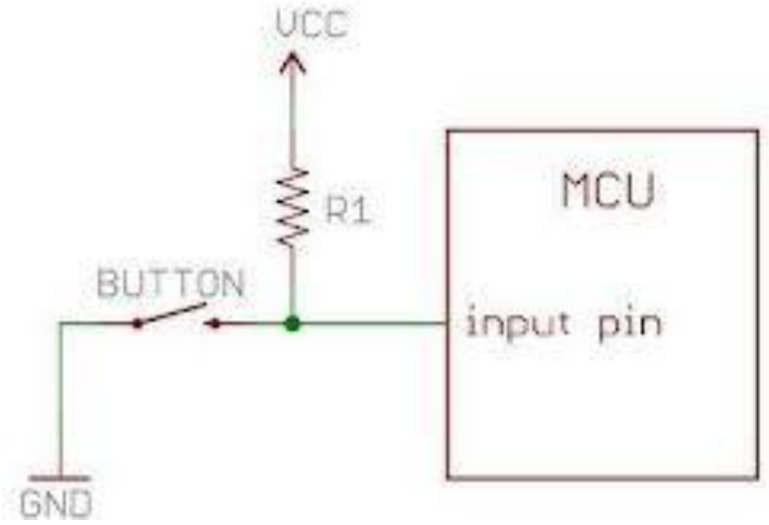
VD: P1DIR = 0x01 //pin P1.0 is output

3.1 Registers: PxIN (x: 1 , 2)

Khi ta cần đọc giá trị từ các chân GPIO input, ta sẽ đọc bit tương ứng trên thanh ghi PxIN

Bit = 0: The input is low
Bit = 1: The input is high

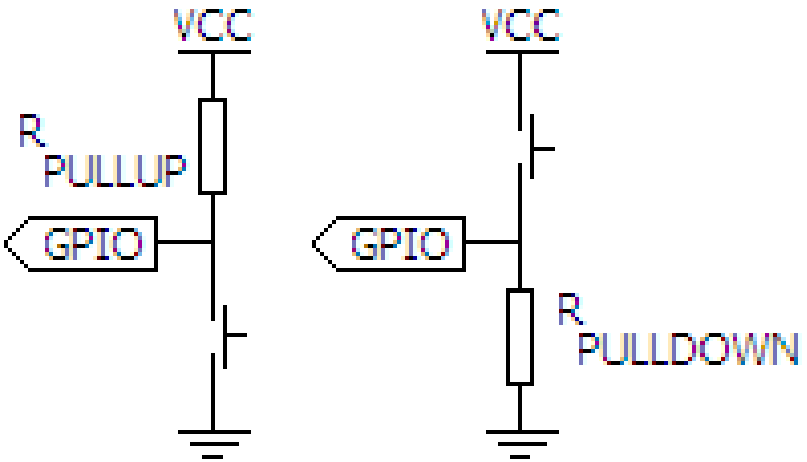
Lưu ý là ta phải chắc rằng bit tương ứng của GPIO trên thanh ghi PxDIR vẫn là 0 (input) trước khi đọc PxIN.

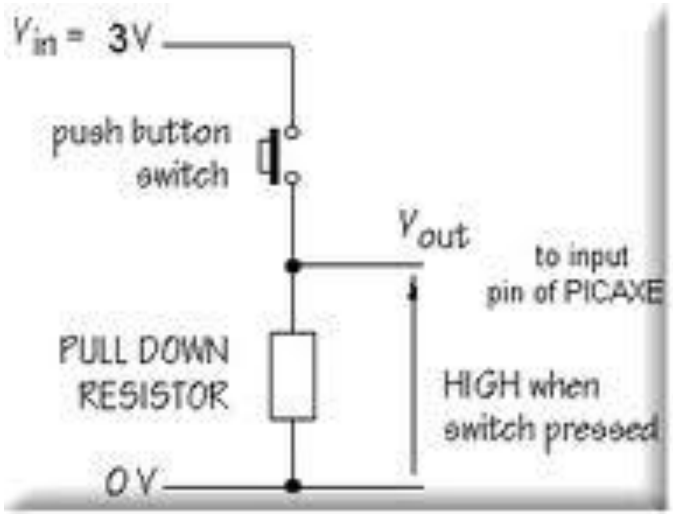


3.1 Registers: PxREN (x:1 , 2)

Thanh ghi này cho phép tắt hoặc mở chế độ pullup/pulldown resistor cho các chân GPIO.
Mặc định các bit của thanh ghi này là 0.

Bit = 0: Pullup/pulldown resistor disabled
Bit = 1: Pullup/pulldown resistor enabled





3.1 Registers: PxOUT (x: 1 , 2)

Khi ta muốn xuất điện áp trên các chân Output, ta sẽ đặt giá trị tương ứng vào các thanh ghi PxOUT (logic 1 -> VCC, logic 0 -> 0V)

Bit = 0: The output is low
Bit = 1: The output is high

Trong trường hợp thanh ghi PxREN qui định GPIO ở chế độ pullup/pulldown resistor thì PxOUT qui định:

Bit = 0: The pin is pulled down
Bit = 1: The pin is pulled up

Lưu ý là ta phải chắc rằng bit tương ứng của GPIO trên thanh ghi PxDIR được set là 1 (output) trước khi xuất mức logic ra.

3.1 Registers: Summary

Để thực hiện xuất / nhập trên 1 chân (Pin):

- Đặt chức năng của pin là GPIO hay các chức năng khác qua thanh ghi PxSEL
- Đặt đúng giá trị cho thanh ghi PxDIR: 1:output; 0: input
- **Xuất** tín hiệu bằng cách ghi giá trị "0" hoặc "1" vào thanh ghi OUT của port tương ứng (vd: P1OUT)
- **Đọc** giá trị của chân bằng cách đọc mức logic trên thanh ghi IN của port tương ứng (vd: P1IN)
- Khi cần điện trở kéo lên/xuống, đặt đúng giá trị của bit trên các thanh ghi PxREN và PxOUT

3.2 Programming:

```
#include <msp430g2553.h>
                                    // hay <msp430.h>
//Khai báo biến toàn cục
unsigned int a

//Khai báo chương trình con
void chuong_trinh_con(void)
{
//chương trình con ở đây
}
```

3.2 Programming:

```
//Chương trình chính
void main(void)
{
        // Phần khởi tạo

        // vòng lặp vô tận
        while(1)
        {
                //thực hiện lệnh
        };

}
```

3.2 Programming: BIT MASK

MASKING BITS TO 1

```
    10011101    10010101
OR  00001000    00001000
 =  10011101    10011101
```

MASKING BITS TO 0

```
     10011101    10010101
AND  11110111    11110111
  =  10010101    10010101
```

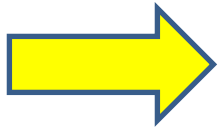
QUERYING THE STATUS OF A BIT

```
     10011101    10010101
AND  00001000    00001000
  =  00001000    00000000
```

TOGGLING BIT VALUES

```
     10011101    10010101
XOR  00001111    11111111
  =  10010010    01101010
```

Bit Mask Defined in MSP430G2553.h

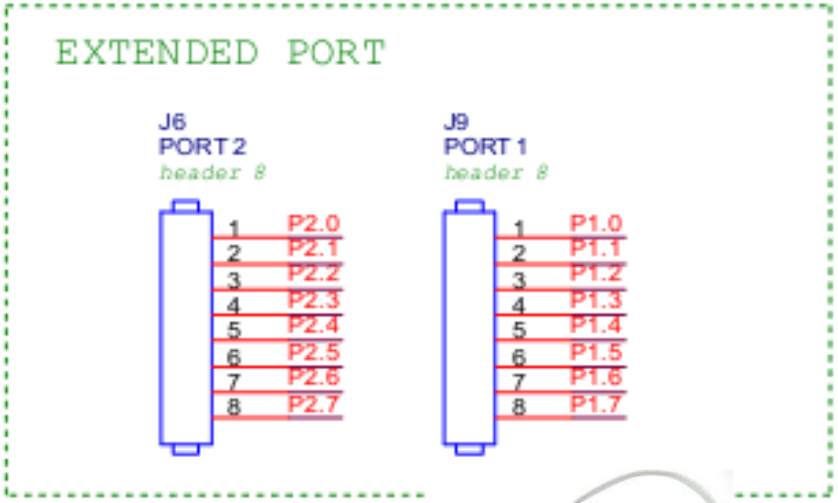


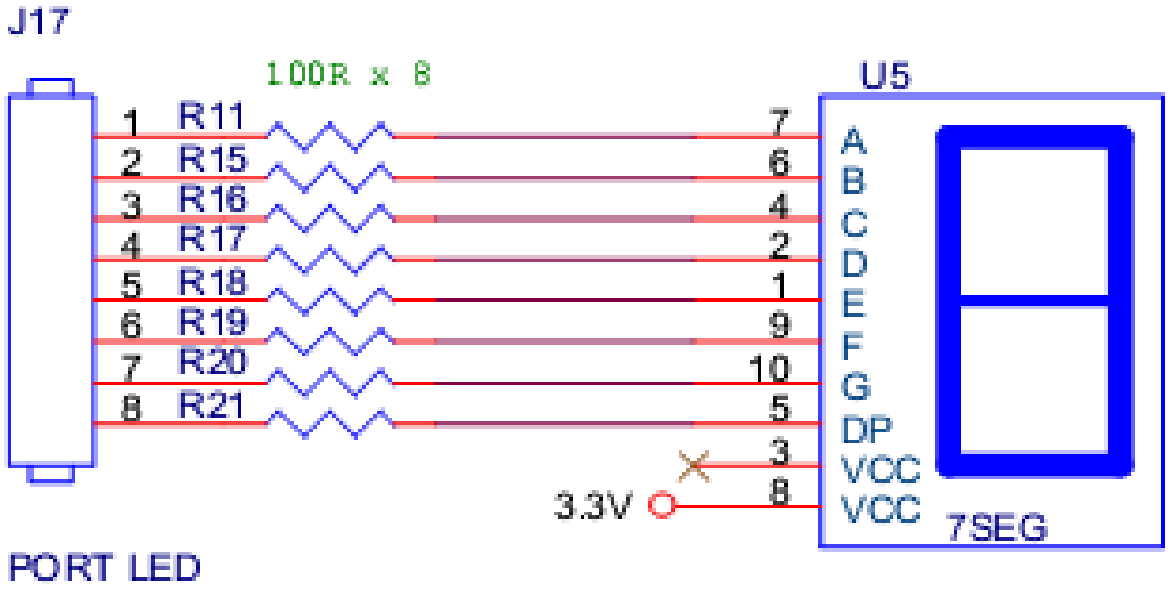
```
#define BIT0        (0x0001)
#define BIT1        (0x0002)
#define BIT2        (0x0004)
// ..............   ..............
#define BITE        (0x4000)
#define BITF        (0x8000)
```

3.3 Ex: Blink a led

Pins: low level --> LEDs On

Pins: high level --> LEDs Off





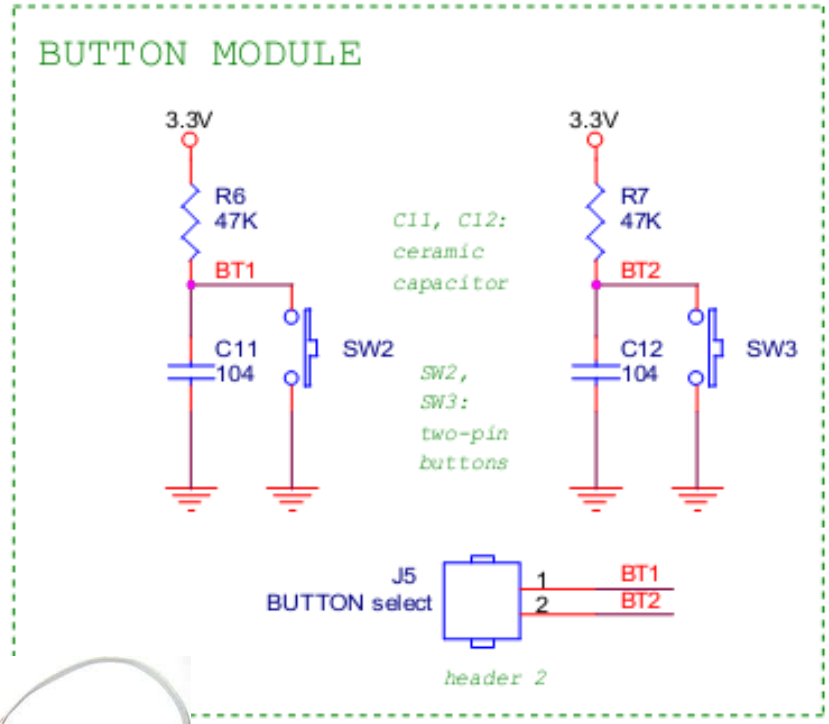
3.3 Ex: Blink a led

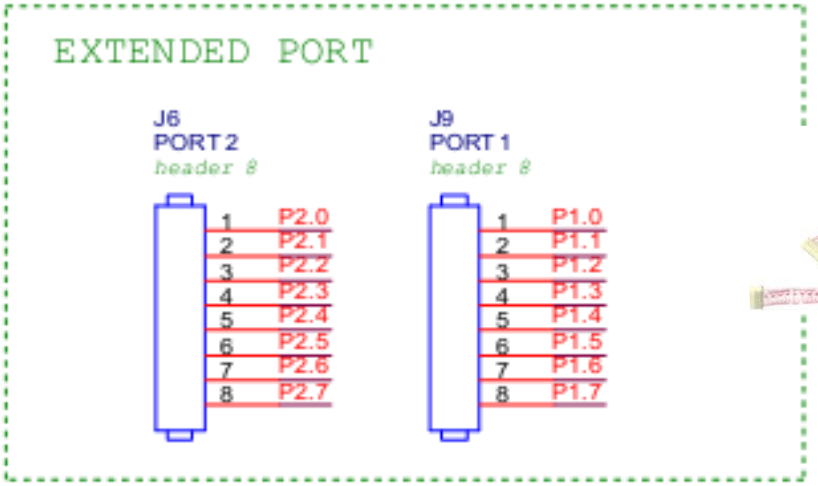
//Blink a led connected with P1.0

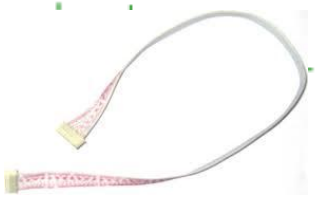
```
#include <msp430g2553.h> // or <msp430.h>
void main(void)
{
// unsigned int i; // declare variables
WDTCTL = WDTPW + WDTHOLD; // Stop watch dog timer
P1DIR |= BIT0 + BIT6;     // Set P1.0 and P1.6 to output
                          // direction

while(1)
{
P1OUT ^= BIT0 + BIT6; // Toggle P1.0 and P1.6 using exclusive-OR
__delay_cycles(1000000); // Delay 1000000 Machine Cycles (MC=1us)
                         // in IAR: _delay_cycles(1000000);
}
}
```

3.4 Ex2: Read a button

Button	BTx voltage	BTx logic
Normal	VCC	1
Pushed	GND	0







3.4 Ex2: Read a button

//Change logic output of P1.0&P1.6 while pressing P1.3

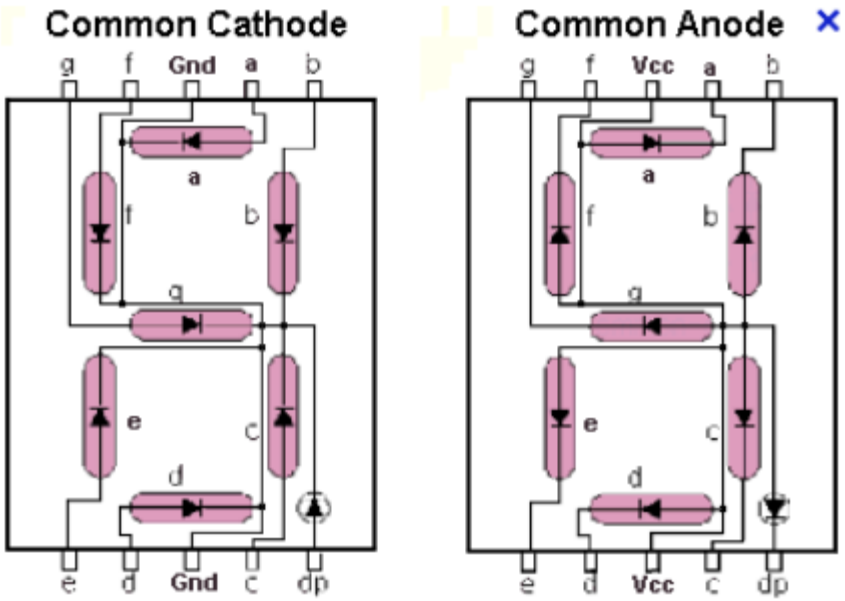
```
#include <msp430g2553.h> // or <msp430.h>
void main(void)
{
WDTCTL = WDTPW + WDTHOLD; // Stop watch dog timer
P1DIR |= BIT0 + BIT6;     // Set P1.0 and P1.6 to output direction

P1DIR &= ~BIT3;      // Set P1.3 to input direction
P1REN |= BIT3;      // Set P1.3 Pull up/Pull down Resistor Enable
P1OUT |= BIT3;      // Set Pull up Resistor at P1.3

while(1)
{
if (P1IN & BIT3) // Check the status of P1.3 (==0 if button is pushed)
P1OUT |= BIT0 + BIT6; // Set P1.0 & P1.6 to high output
else
P1OUT &= ~(BIT0 + BIT6); // Set P1.0 & P1.6 to low output
}
}
```

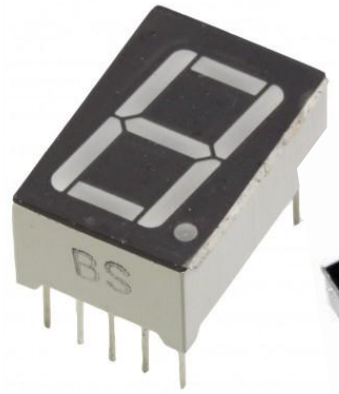
Part 4:
Discussion & Homework

Homework: optional

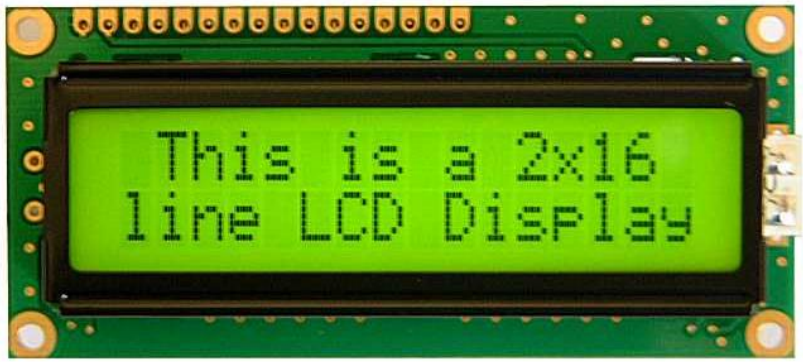
- LED 7 Segment
- LCD 16x2
- LED Matrix 8x8
- Keypad 4x4

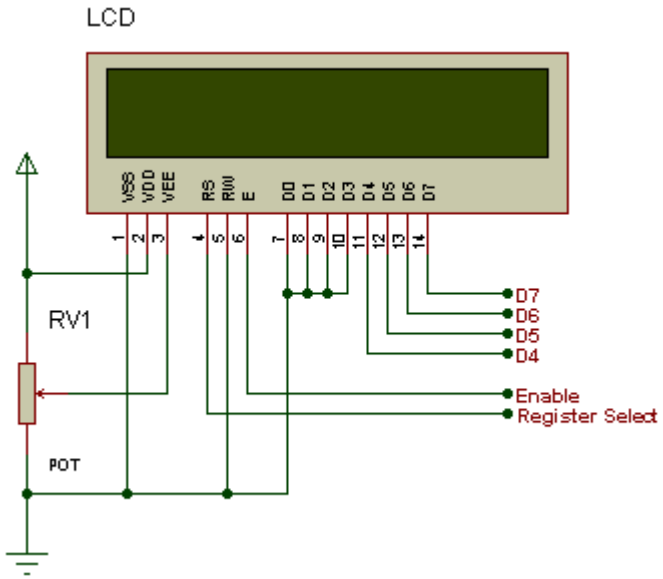
Module: Led 7 segments

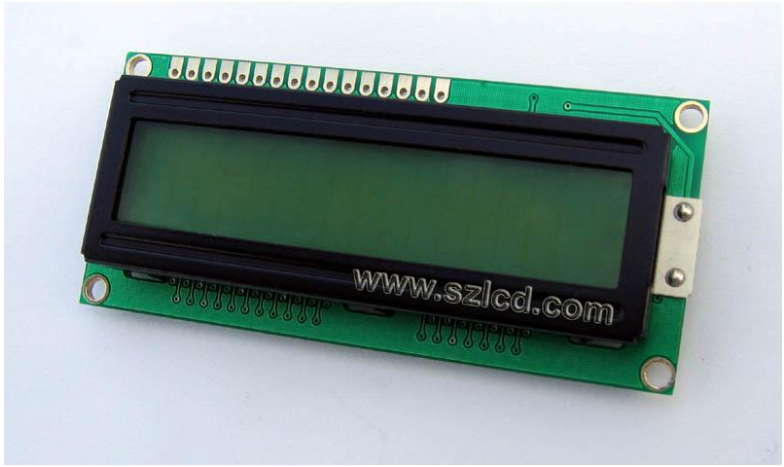




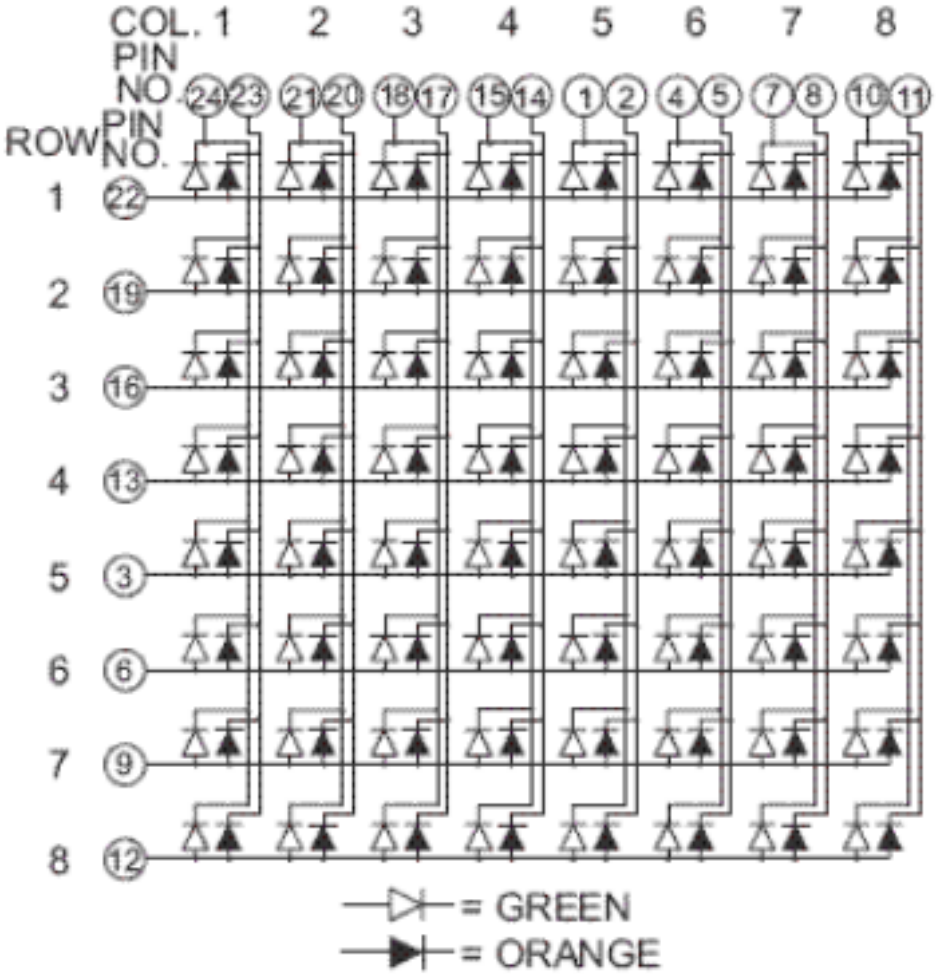
Module: LCD

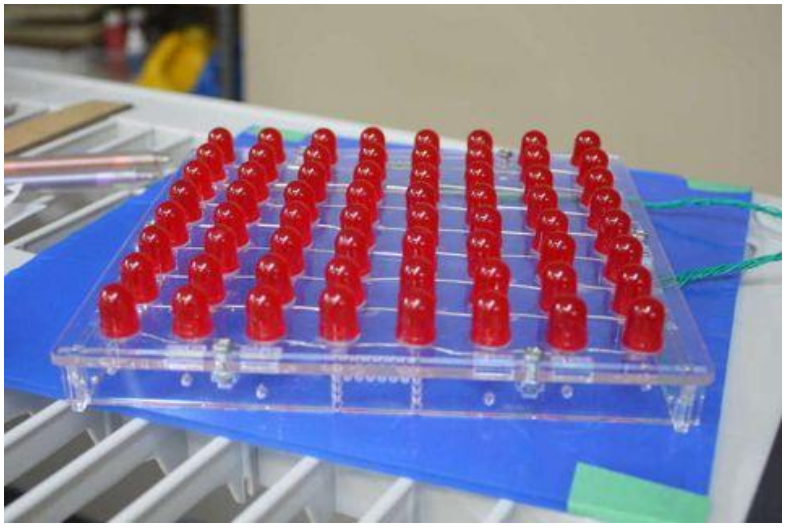

This is a 2x16
line LCD Display

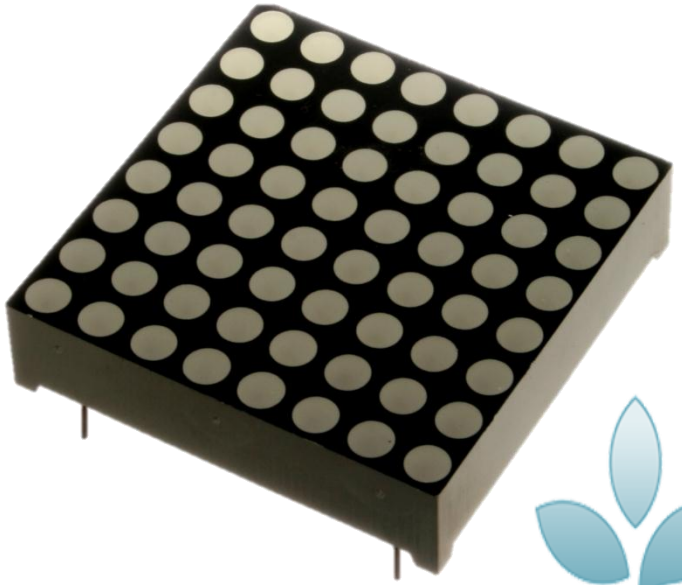

LCD
VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
1
2
3
4
5
6
7
8
9
10
11
12
13
14
RV1
POT
D7
D6
D5
D4
Enable
Register Select


www.szlcd.com

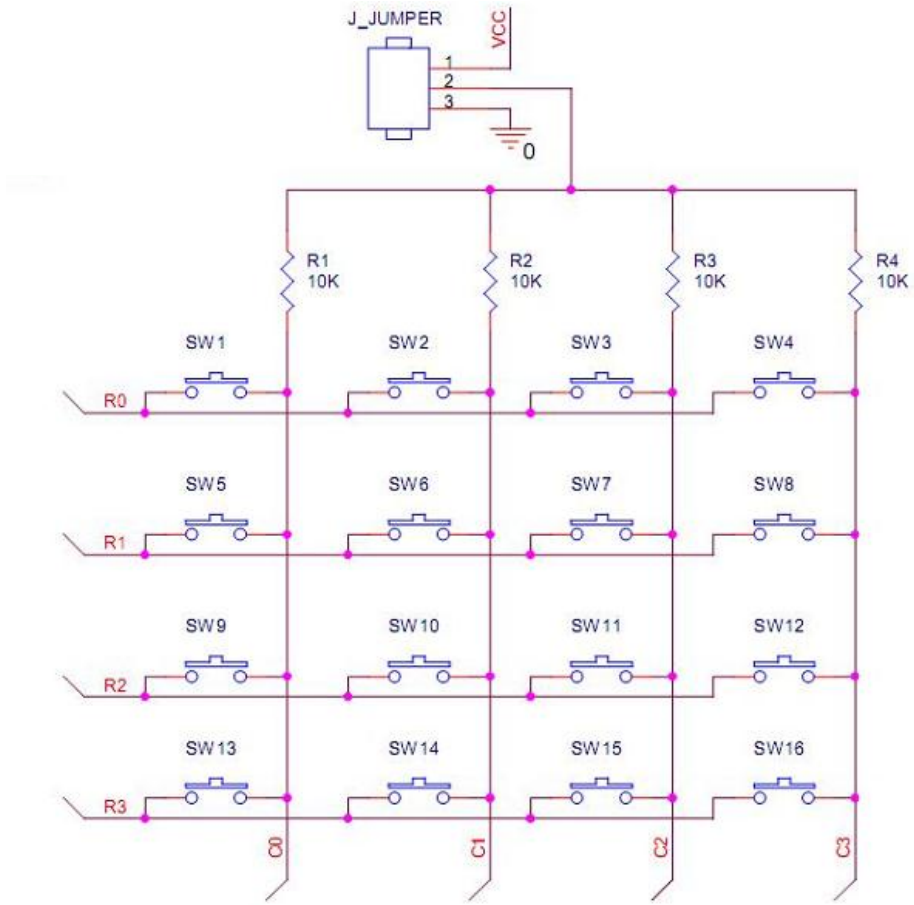
Module: Led Matrix







Module: Keypad



PAY IT FORWARD

payitforward.edu.vn