POLITEKNIK MALAYSIA

KEMENTERIAN PENGAJIAN TINGGI

KOLEJ KOMUNITI MALAYSIA

PIC18 BASED PROJECTS COLLECTION

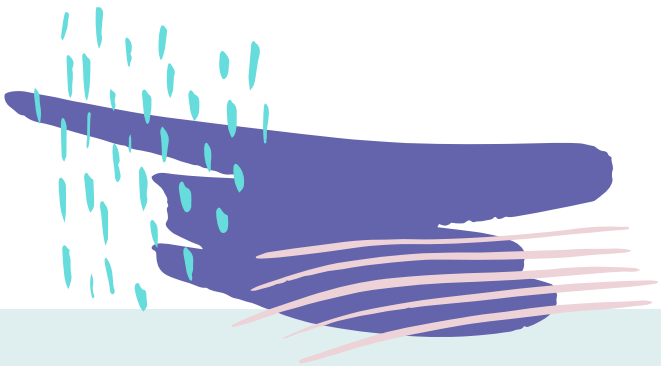
TUTORIAL STEP BY STEP

ZUNAINAH BINTI HAMID
INTAN SHAFINAZ BINTI ABD. RAZAK
KHAIROL SHAH BIN OTHMAN

ELECTRICAL ENGINEERING DEPARTMENT

TABLE OF

CONTENTS



Copyright

PIC18 Based Projects Collection.
Tutorial Step By Step

Authors:
Zunainah Binti Hamid
Intan Shafinaz Binti Abd Razak
Khairol Shah Bin Othman

Politeknik Tuanku Sultanah Bahiyah (PTSB),
Kulim Hi-Tech Park, 09090 Kulim, Kedah

ISBN 9789672740117

Preface
About This Book

This ebook is written to help students understand the basic skills in designing an embedded system using PIC18. There are five examples of applications embedded in real life.

The first, second, and third examples in this lesson use seven (7) methods each to help students comprehend the entire process in PIC programming.

The last two examples merely provide a simple strategy for resolving PIC programming.

Hopefully, students will learn how to design and program code in a straightforward and concise manner.

Acknowledge

Thank You Allah for his mercy and blessing who gave us the strength and patient to finish our e-book. Our appreciation also goes to PTSB E-book Organizer all colleagues for their cooperation and guidance during completing our first PIC18 Based Project Collection ebook.

introduction

Why PIC programming is good for you?

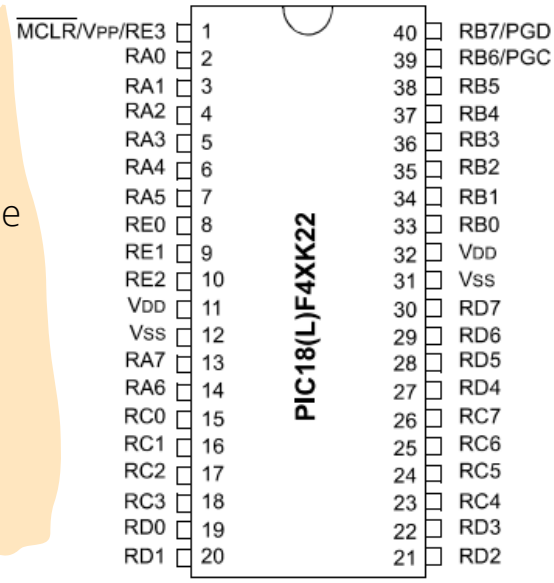
- [ ] Increase creativity
- [ ] Always learning
- [ ] Develop self-disciplined
- [ ] Logical mindset
- [ ] Planning skill
- [ ] Adaptable and dynamic
- [ ] Documentation

yes !
code your mindset

CHEAT SHEET
ABOUT PIC18

PORTx

A special function register (in Data Memory)
The content of this register corresponds to the value of the electrical signal at its IO pin
Reads the device level
Store the input level of pins
Reads the input signal from external (if the pin is configured as Input)
"1" results high voltage
"0" results low voltage



Pin B0 to B7 are HIGH

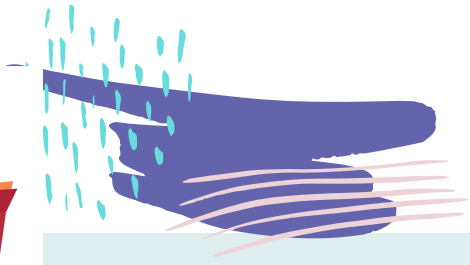
```
PORTB = 0b1111111;

PORTB= 255;

PORTB= 0xFF;
```

Pin B0, B1, B2, B3 are HIGH, B4, B5, B6 & B7 are LOW

```
PORTB = 0b00001111;

PORTB= 15;

PORTB= 0x0F;
```

Pin B0 is HIGH

```
PORTBbits.RB0 = 1;
```



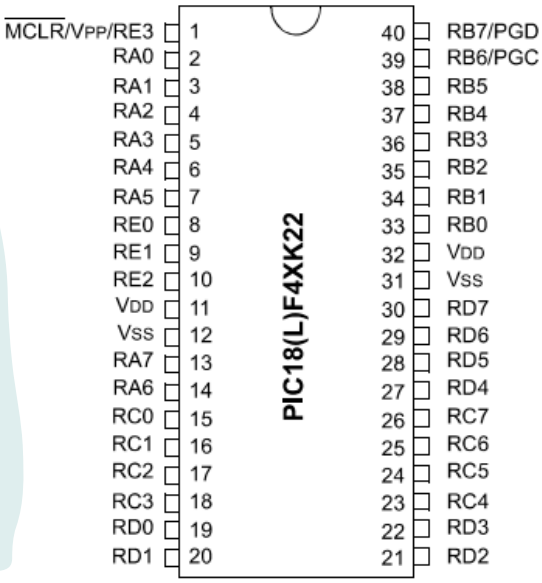
CHEAT SHEET
ABOUT PIC18

TRISx

Each IO pin's direction (input or output) is controlled by TRIS.

Tri-state, which refers to the possible states of a pin, is simplified into the word TRIS. output logic 1 and output logic 0

That pin functions as an output if the bit is set to 0. A pin is an input if a bit is set to 1. Because a 0 represents an O for output and a 1 matches an I for input, this is simple to remember.



Pin B0 to b7 are INPUT

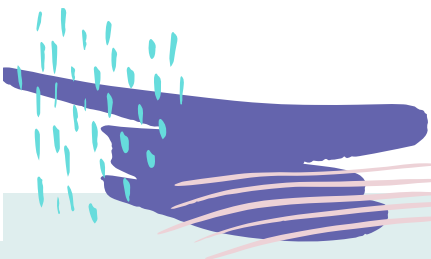
```
TRISB= 0b1111111;

TRISB= 255;

TRISB= 0xFF;
```

Pin B0, B1, B2, B3 are INPUT, B4, B5, B6 & B7 are OUTPUT

```
TRISB= 0b00001111;

TRISB= 15;

TRISB = 0x0F;
```

Pin B0 is INPUT

```
TRISBbits.TRISB0 = 1;
```



CHEAT SHEET
ABOUT PIC18

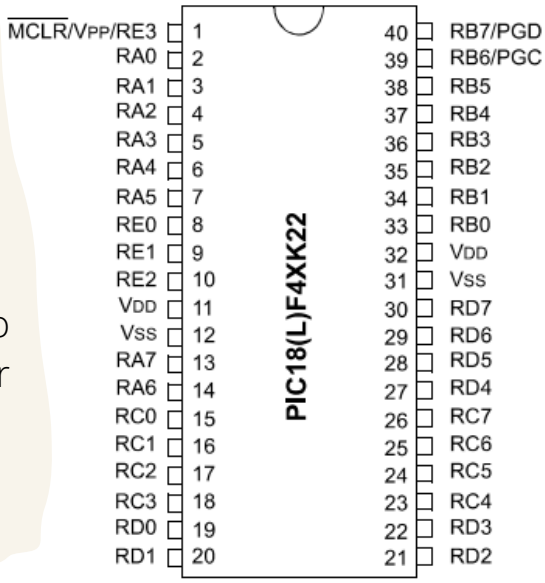
LATx

LAT register (output latch)

When a pin is assigned to output, the LAT register will control the output logic state of that pin.

Logic High (VDD) is made by putting the LAT register to 1 and Logic Low (A) is made by putting the LAT register to 0. (VSS).

The LAT register bits for Input pins are Don't Care since they have no effect on either Input or Output.



Pin B0 to B7 are at logic HIGH

LATB= 0b1111111;

LATB= 255;

LATB= 0xFF;

Pin B0, B1, B2, B3 are at logic HIGH, B4, B5, B6 & B7 are at logic LOW

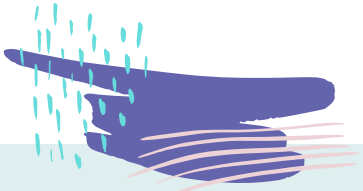
LATB= 0b00001111;

LATB= 15;

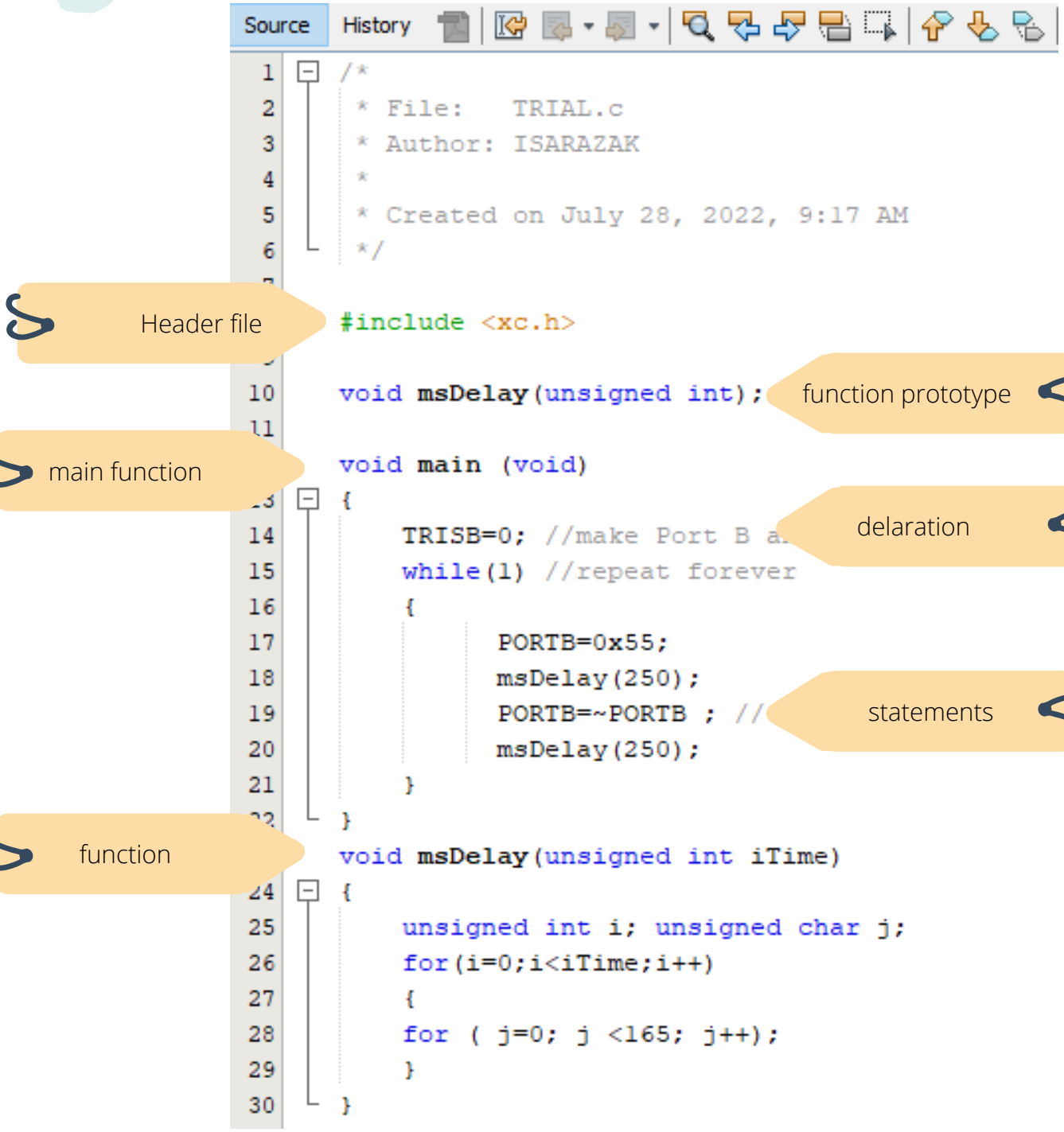
LATB = 0x0F;

Pin B0 at logic HIGH

LATBbits.LATB0 = 1;



UNDERSTAND
BASIC STRUCTURE OF C PROGRAM



7 STEPS

PROBLEM STATEMENT

CIRCUIT DESIGN -MAIN COMPONENT & CIRCUIT

CIRCUIT DESIGN - FLOW PROCESS

CONFIGURE INPUT & OUTPUT

PROGRAM IT USING C LANGUAGE

EXPLORE THE DATASHEET

ENHANCE - UNDERSTANDING

01

project 01

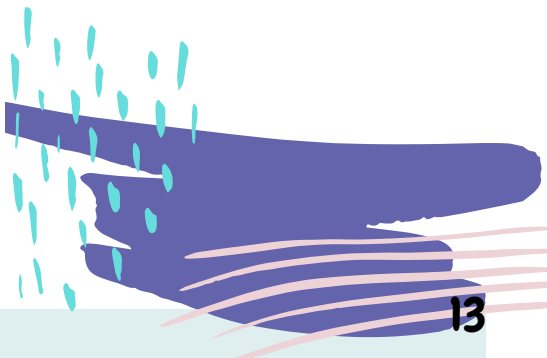
1.PROBLEM STATEMENT

A recycle factory needs an automatic system to isolate three types of bottle based on its height. Ultrasonic sensor is use to detect the height. Ultrasonic sensor is use to detect the heigt of the bottles. Bottle categories have summarized as shown in Table below. If bottle in Category 1 detected, conveyor 1 will running to move the bottle for isolation. While, if a bottle in Category 2 detected, the conveyor 2 will be running. Otherwise, if the bottle in Category 3, Conveyor 3 will be running. With the information as in Diagram B2, design the appropriate system.

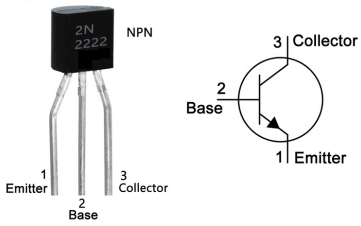
SPECIFICATION:

VREF(+)=5V
VREF(-)=0V
SENSOR SCALE FACTOR = 9.766 mV/cm
using Channel AN0
A/D conversion results format: Left justified
A/D Conversion Clock FOSC/16
12 TAD acquisition time
Resolution = 10 bits

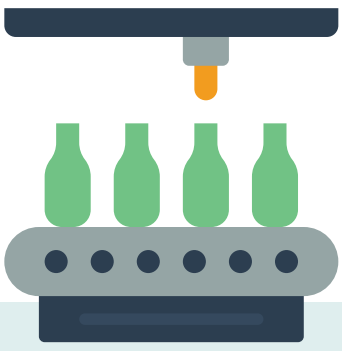
CATEGORY	HEIGHT,cm
1	less than 10
2	between 10 to 20
3	more than 20

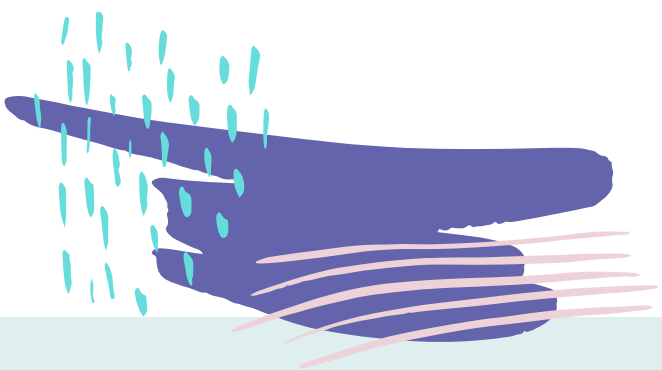


2. CIRCUIT DESIGN
MAIN COMPONENT



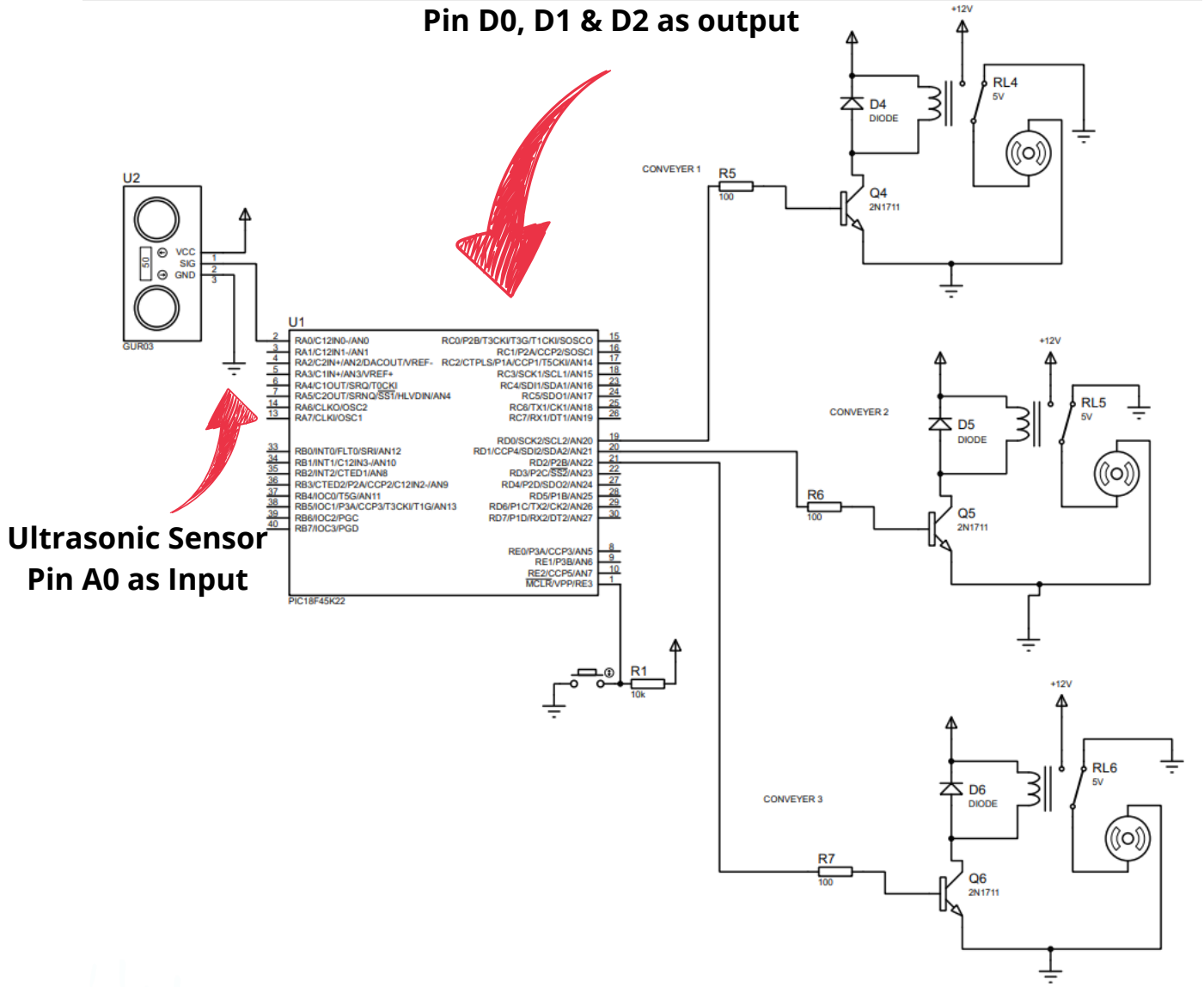
- [x] PIC 18 F45K22
- [x] Ultrasonic Sensor
- [x] 3 units motor (as conveyor)
- [x] 3 units diode
- [x] 3 units 1 channel relay module
- [x] 3 units transistor 2N1711
- [x] 12V and 5V supply



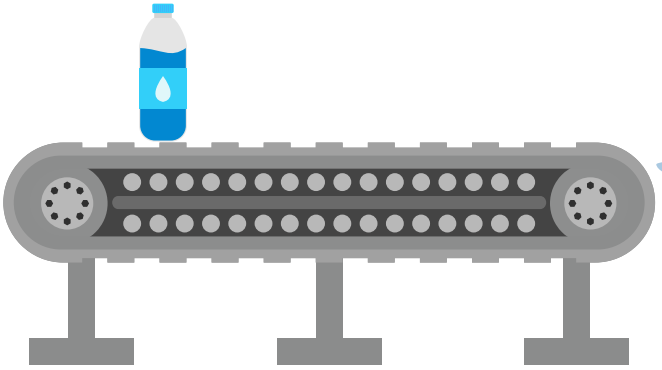


2. DESIGN
SCHEMATIC CIRCUIT

DRAW THE CIRCUIT:

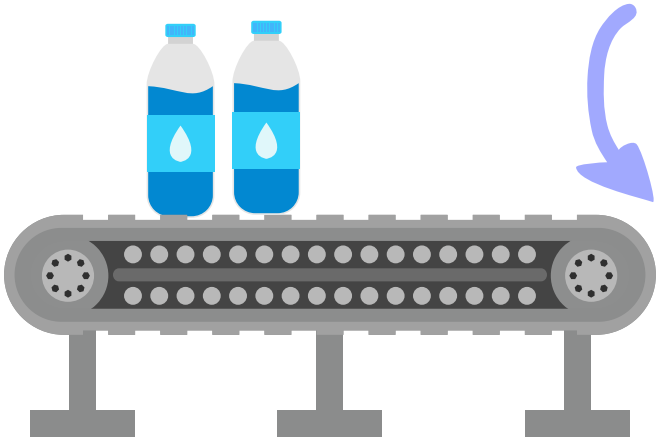


3. CIRCUIT DESIGN
FLOW PROCESS

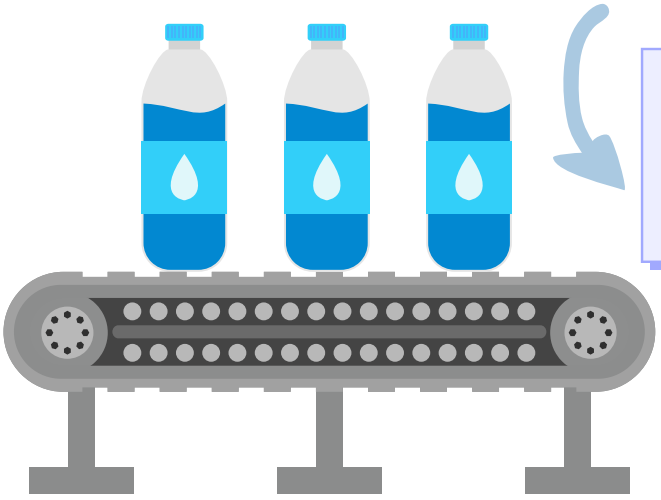


detect height of bottle

if bottle is less than 10 cm, conveyor 1 on

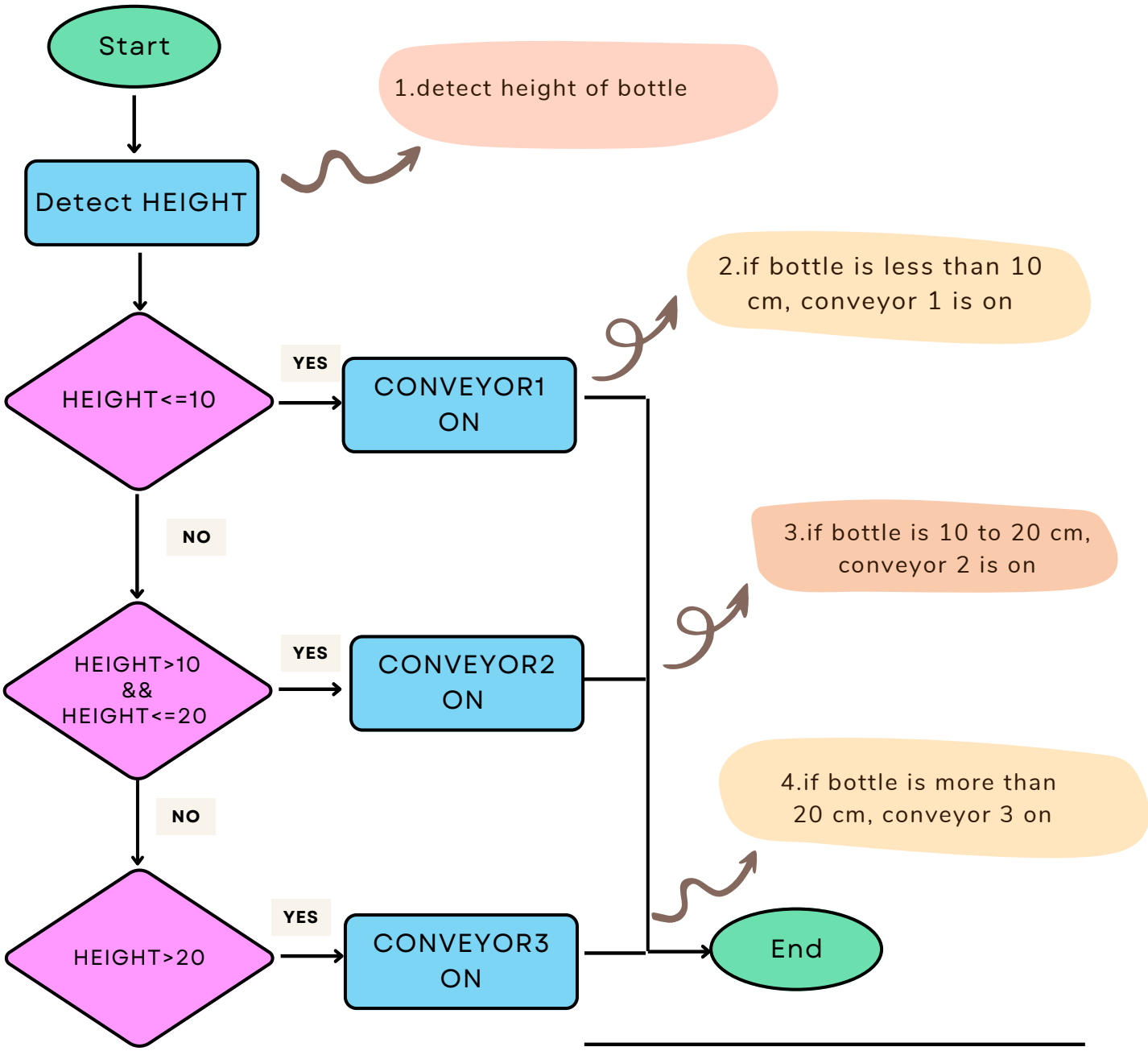


if bottle is 10 to 20 cm, conveyor 2 on

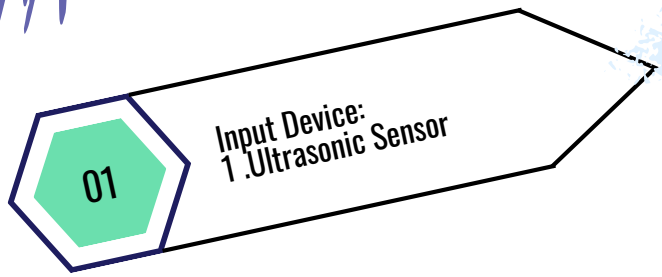


if bottle is more than 20 cm, conveyor 3 on

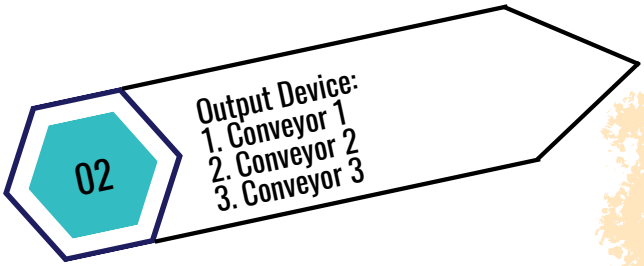
3.UNDERSTAND
FLOW CHART



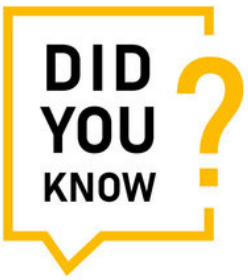
4. CONFIGURATION
INPUT/OUTPUT



analog sensor as Input
Pin A0 as Input
TRISAbits.RA0 = 1;



digital output
Pin D0, D1 & D2 as output
TRISDbits. RD0 = 0;
TRISDbits. RD1 = 0;
TRISDbits. RD2 = 0;



TRISBbits.RB1=1 ;

INPUT = 1 ;

TRISBbits.RB1=0 ;

OUTPUT = 0 ;

We use "TRIS" to configure Input & Output ;

5. PROGRAMMING
C LANGUAGE

CONFIGURE THE ADC MODULE:

01

Select ADC input channel.
Turn on ADC module .
[ADCON0]

02

Configure voltage reference.
[ADCON1] [VREFCON]

03

Select result format. Select acquisition delay.
[ADCON2 & ADRES]

04

Start conversion by setting the GO/DONE bit.
Wait for ADC conversion to complete

05

Read ADC Result

5. PROGRAMMING
C LANGUAGE

MPLAB SIMULATION

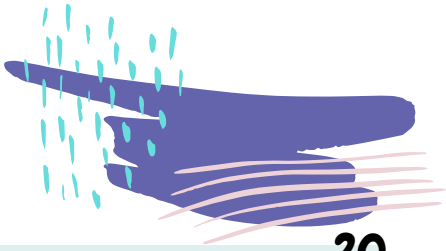
```
/*
 * File:   3conveyor.c
 * Author: IntanShafinazAbdRazak
 *
 * Created on May 30, 2022, 10:05 AM
 */

//HEADER FILE
#include <xc.h>
#define _XTAL_FREQ 64000000
#define CONVEYOR1 LATDbits.LATD0
#define CONVEYOR2 LATDbits.LATD1
#define CONVEYOR3 LATDbits.LATD2
//GLOBAL DECLARATION
float ADRESULT;
float HEIGHT;

//MAIN FUNCTION
void main(void)
{
//PORT INITILIZATION
TRISD=0x00;
TRISAbits.RA0=1;
//ADC SET UP
ANSELA=0b00000001;
ANSELD=0b00000000;
ADCON0=0b00000001;
ADCON1=0b00001110;
ADCON2=0b00101101;
```

declare CONVEYOR1, 2 & 3 using #define

ADRESULT & HEIGHT declared as Floating number (due to decimal point in the **formula***)

Pin initialization, Pin D0-D7 as output and Pin A0 as input.

Analog/Digital Conversion set up. Please refer to **datasheet****



5. PROGAMMING
C LANGUAGE

MPLAB SIMULATION

```
while(1)
{
    ADCON0bits.GO_DONE=1; //ADC ENABLE
    while (ADCON0bits.GO_DONE==0); //COMPLETED
    ADRESULT =ADRESH << + ADRESL;
    HEIGHT= ADRESULT * 5/(1024*0.009766);

    if (HEIGHT <=10)
    { CONVEYOR1 = 1;
    CONVEYOR2 = 0;
    CONVEYOR3 = 0;
    }

    if ((HEIGHT > 10)&&(HEIGHT <=20))
    {CONVEYOR1 = 0;
    CONVEYOR2 = 1;
    CONVEYOR3 = 0;
    }

    if(HEIGHT > 20)
    {CONVEYOR1 = 0;
    CONVEYOR2 = 0;
    CONVEYOR3 = 1;
    }
}
}
```

GO_DONE=1, enable the Analog/ Digital Conversion

GO_DONE==0, means check status of Analog/Digital Conversion, it is completed?

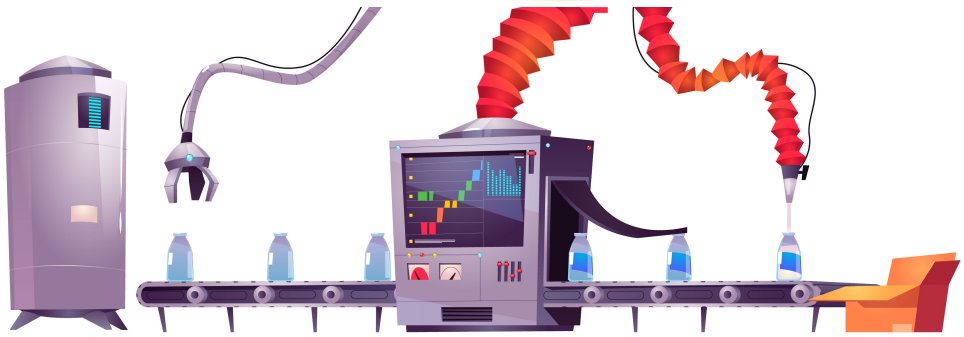
Keep ADRESULT to Left Justified in ADRES register (shift to ADRESH then add LSB to ADRESL)

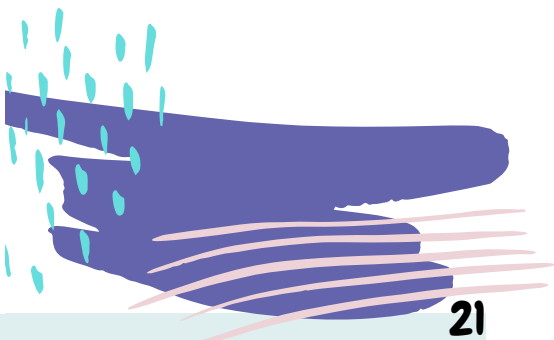
formula*

$$\text{HEIGHT} = \frac{\text{RESULT X VREF}}{\text{RESOLUTION X SENSOR SCALE FACTOR}}$$

VREF(+)=**5**V
VREF(-)=0V
SENSOR SCALE FACTOR = 9.766 mV/cm
So, SENSOR SCALE FACTOR= **0.009766** mV/cm
using Channel AN0
A/D conversion results format: Left justified
A/D Conversion Clock FOSC/16
12 TAD acquisition time
Resolution = 10 bits =2^10=**1024**

If statements for 3 conditions of HEIGHT





6. REGISTER
EXPLORE DATASHEET

ADCON0= 0b0**0000001**;

REGISTER 17-1: ADCON0: A/D CONTROL REGISTER 0

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CHS<4:0>					GO/$\overline{\text{DONE}}$	ADON
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

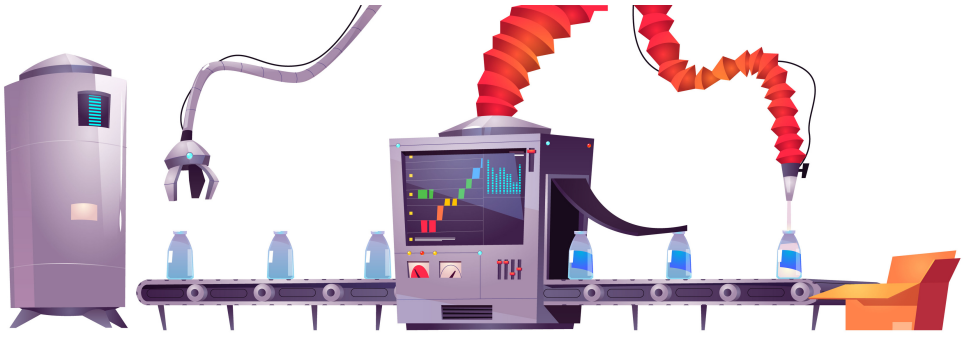
bit 7 **Unimplemented:** Read as '0'

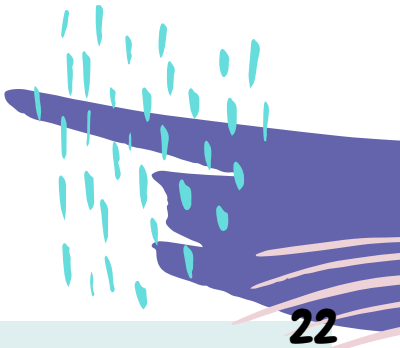
bit 6-2 **CHS<4:0>: Analog Channel Select bits**
00000 = AN0
00001 = AN1
00010 = AN2
00011 = AN3
00100 = AN4

bit 1 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
1 = A/D conversion cycle in progress. Setting this bit starts an A/D conversion cycle.
This bit is automatically cleared by hardware when the A/D conversion has completed.
0 = A/D conversion completed/not in progress

bit 0 **ADON:** ADC Enable bit
1 = ADC is enabled
0 = ADC is disabled and consumes no operating current

Note 1: Available on PIC18(L)F4XK22 devices only.
2: Allow greater than 15 µs acquisition time when measuring the Fixed Voltage Reference.





6. REGISTER
EXPLORE DATASHEET

ADCON1= 0b**0**0001**1110**;

REGISTER 17-2: ADCON1: A/D CONTROL REGISTER 1

R/W-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
TRIGSEL	—	—	—	PVCFG<1:0>		NVCFG<1:0>	
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

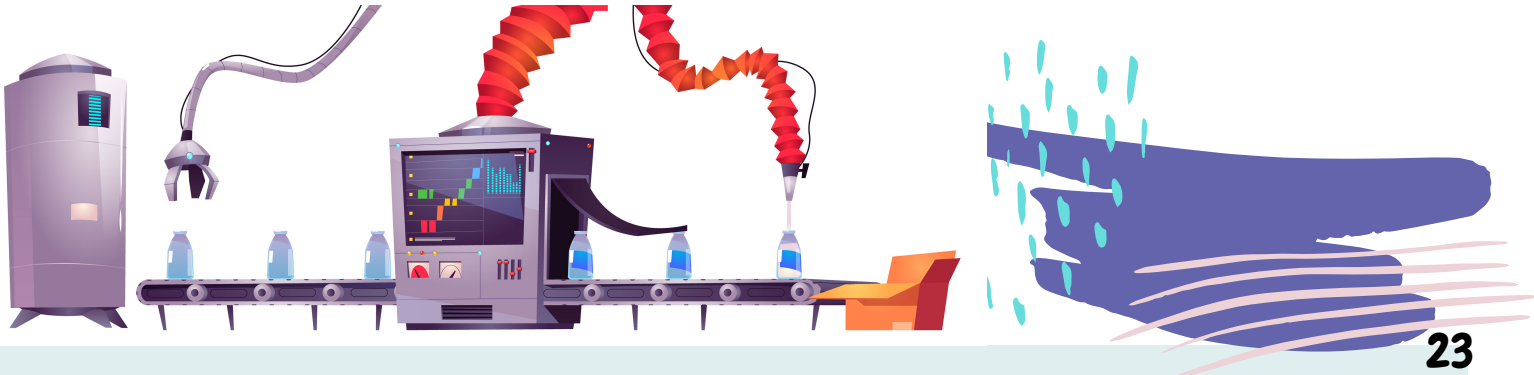
bit 7 **TRIGSEL**: Special Trigger Select bit
1 = Selects the special trigger from CTMU
0 = Selects the special trigger from CCP5

Additional Function on
- CCP5 will
  - Set ADCON0<1>, GO/$\overline{\text{DONE}}$ bit to start an ADC Conversion if ADCON<0>, ADON = 1.

bit 6-4 **Unimplemented:** Read as '0'

bit 3-2 **PVCFG<1:0>:** Positive Voltage Reference Configuration bits
00 = A/D VREF+ connected to internal signal, AVDD
01 = A/D VREF+ connected to external pin, VREF+
10 = A/D VREF+ connected to internal signal, FVR BUF2
11 = Reserved (by default, A/D VREF+ connected to internal signal, AVDD)

bit 1-0 **NVCFG<1:0>:** Negative Voltage Reference Configuration bits
00 = A/D VREF- connected to internal signal, AVSS
01 = A/D VREF- connected to external pin, VREF-
10 = Reserved (by default, A/D VREF- connected to internal signal, AVSS)
11 = Reserved (by default, A/D VREF- connected to internal signal, AVSS)



6. REGISTER
EXPLORE DATASHEET

ADCON2= 0b**0**0**1011101**;

REGISTER 17-3: ADCON2: A/D CONTROL REGISTER 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT<2:0>			ADCS<2:0>		
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

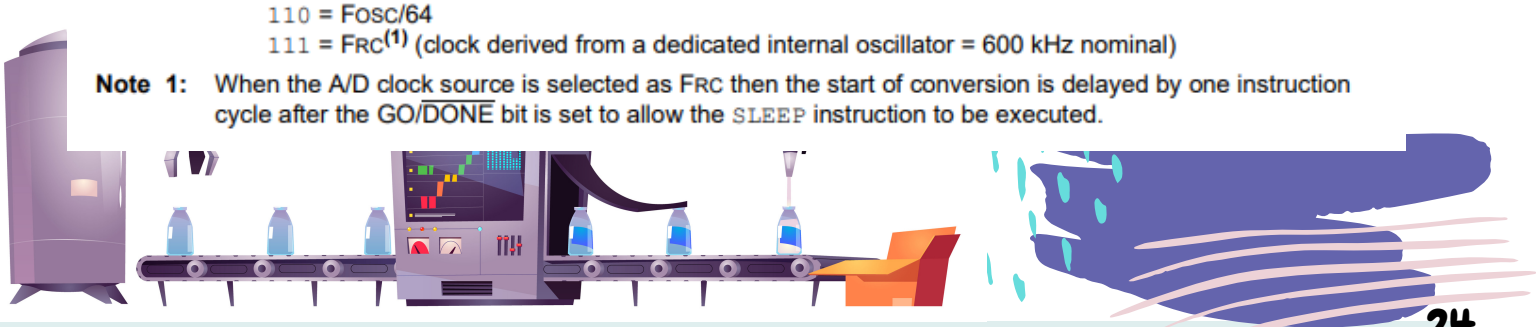
bit 7 **ADFM:** A/D Conversion Result Format Select bit
1 = Right justified
0 = Left justified

bit 6 **Unimplemented:** Read as '0'

bit 5-3 **ACQT<2:0>:** A/D Acquisition time select bits. Acquisition time is the duration that the A/D charge holding capacitor remains connected to A/D channel from the instant the GO/$\overline{\text{DONE}}$ bit is set until conversions begins.
000 = 0(1)
001 = 2 TAD
010 = 4 TAD
011 = 6 TAD
100 = 8 TAD
101 = 12 TAD
110 = 16 TAD
111 = 20 TAD

bit 2-0 **ADCS<2:0>:** A/D Conversion Clock Select bits
000 = FOSC/2
001 = FOSC/8
010 = FOSC/32
011 = FRC(1) (clock derived from a dedicated internal oscillator = 600 kHz nominal)
100 = FOSC/4
101 = FOSC/16
110 = FOSC/64
111 = FRC(1) (clock derived from a dedicated internal oscillator = 600 kHz nominal)

Note 1: When the A/D clock source is selected as FRC then the start of conversion is delayed by one instruction cycle after the GO/$\overline{\text{DONE}}$ bit is set to allow the SLEEP instruction to be executed.



7. ENHANCE
UNDERSTANDING

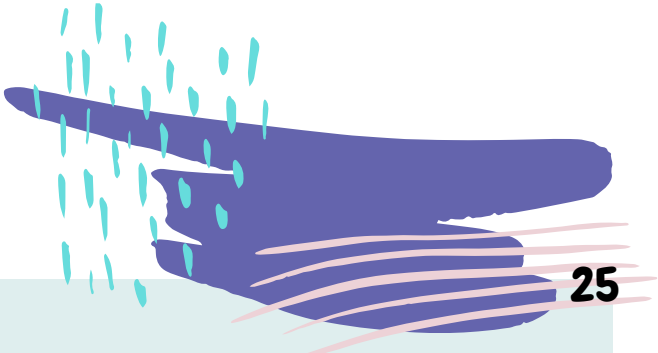
Now, with the same situation, we will change the control system to the following situation.

- A factory needs an automatic system to run a cooling system based on temperature in their control room. Temperature sensor (LM35) is use to detect the temperature. The temperature control conditions have summarized as shown in Table below.
- With the information as in the specification below, design the appropriate system.
- Build the complete program & circuit of the control system.

SPECIFICATION:

VREF(+)=Fixed voltage references (4.096V)
VREF(-)=0V
SENSOR SCALE FACTOR = 10.0 mV/ degree Celcius
using Channel AN2
A/D conversion results format: Right justified
A/D Conversion Clock FOSC/16
12 TAD acquisition time
Resolution = 10 bits

degree Celcius	output
0 - 20	fan 1 on
21 -35	fan 1 & 2 on
> 35	fan 1,2,3 & buzzer on



NOTES

/ /

S M T W T F S

TAKE A BREAK

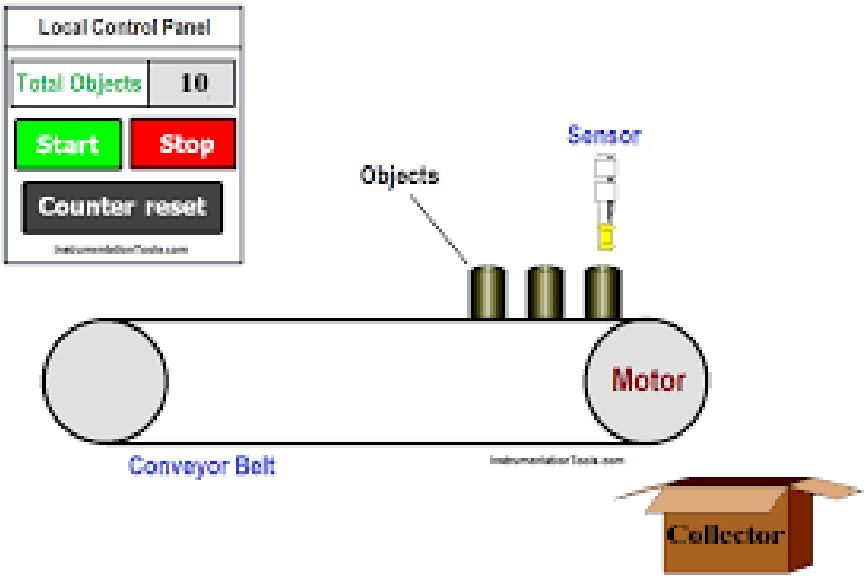
I am fueled by happy thoughts and coffee

02

project 02

1.PROBLEM STATEMENT

An automation company has been assigned by a food and beverage manufacturing plant to design and install a packaging system for bottle based on PIC18 microcontroller. This system used a motor to move a conveyor belt that carry and drops bottles into a box. A counter sensor will count the amount of bottles that fall into the box. After TEN (10) bottles has fallen, motor will stop moving and buzzer will be switched on. Sensor is ACTIVE LOW. Buzzer and motor are ACTIVE HIGH. As a programmer of the company, you are required to design the packaging system. Using Timer0 Interrupt as a counter, Use a C language to write program to perform.

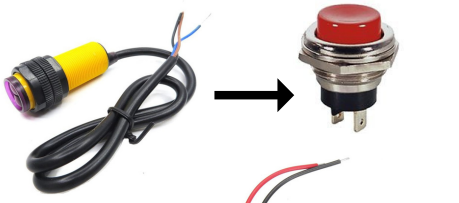


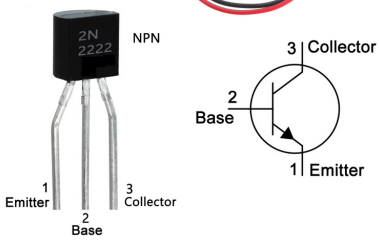
FLOW PROCESS

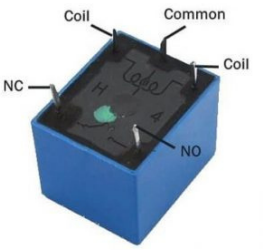
- Conveyer motor moving
- Sensor count 10 objects
- Conveyer motor stop
- Switch on buzzer

2.CIRCUIT DESIGN

MAIN COMPONENT



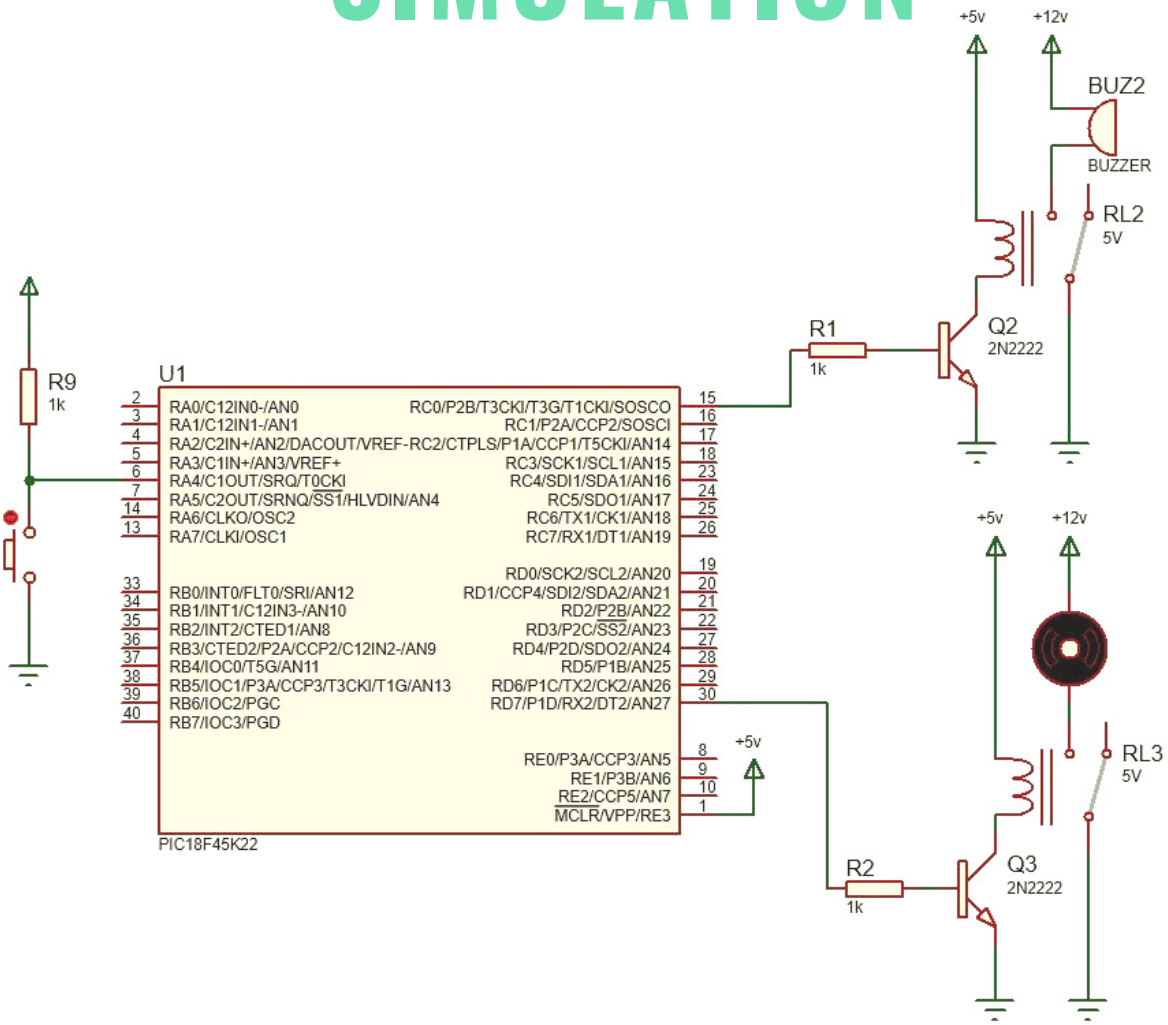


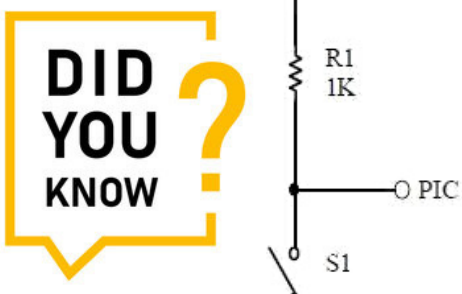


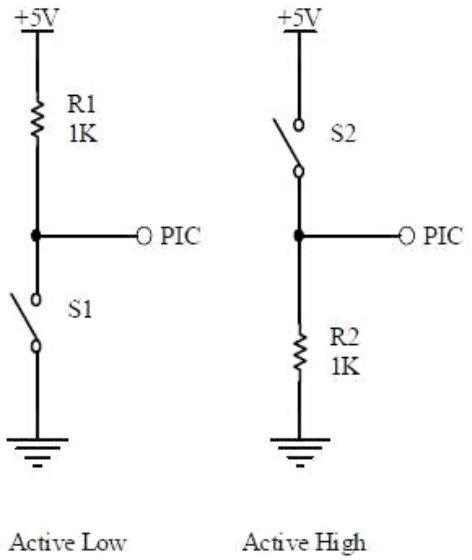
- [x] PIC 18F45K22
- [x] Active push button (sensor)
- [x] Animated DC Motor
- [x] Active buzzer
- [x] Transistor (2N2222)
- [x] 5V relay
- [x] 1K Ohm resistor

3. CIRCUIT DESIGN

PROTEUS SIMULATION





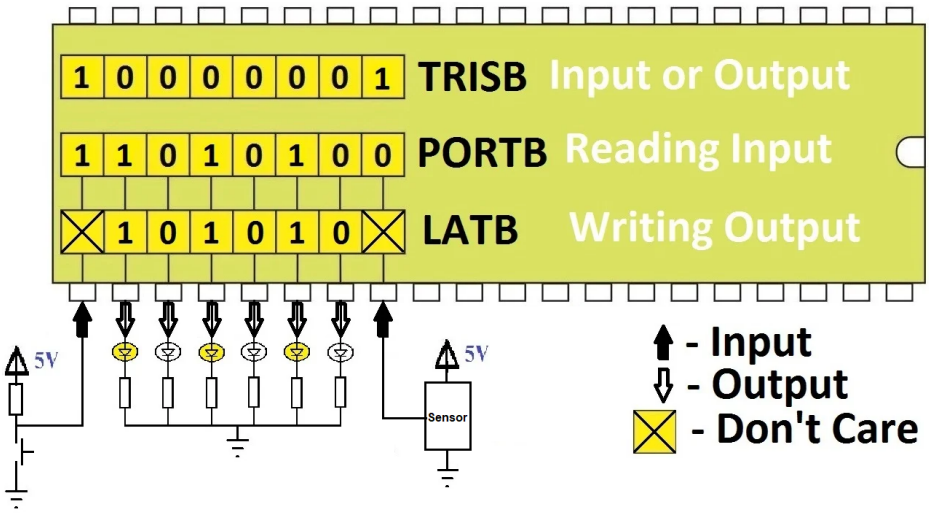


4.CONFIGURATION
INPUT/OUTPUT

Based on the given problem, we can summarize as in the following table.

Component	Reading Input	Writing Output	Input/Output declaration
Push button (sensor)	PORTAbits.RA4	-	TRISAbits.TRISA4
DC Motor (conveyer)	-	LATDbits.LATD7	TRISDbits.TRISD7
Buzzer (alarm)	-	LATCbits.LATC0	TRISCbits.TRISC0

DID YOU KNOW ?



5. PROGRAMMING
C LANGUAGE MPLAB SIMULATION

```
/*
 * File:   newmain.c
 * Author: khashah
 *
 * Created on July 27, 2022, 10:31 PM
 */

#include <xc.h>
#define BUZZER LATCbits.LATC0
#define MOTOR LATDbits.LATD7

void main(void)
{
    TRISAbits.TRISA4=1;
    TRISCbits.TRISC0=0;
    TRISDbits.TRISD7=0;

    T0CON = 0b11111000;
    TMR0L = 0xF6;
    TMR0H = 0;

    BUZZER = 0;
    MOTOR = 1;

    while(1)
    {
        if(INTCONbits.TMR0IF==1)
        {
            BUZZER = 1;
            MOTOR = 0;
            T0CONbits.TMR0ON=0;
            INTCONbits.TMR0IF=0;

        }
    }

}
```

Define variable name

RA4/T0CK1 is counter sensor - input
RC0 connected to buzzer - output
RD7 connected to motor - output

TMR0ON, T08BIT, T0CS,T0SE & PSA all set to 1.
Refer to T0CON register for details.

8 bit counter =256. Therefore 256-10=246.
(246)10 = (F6)16

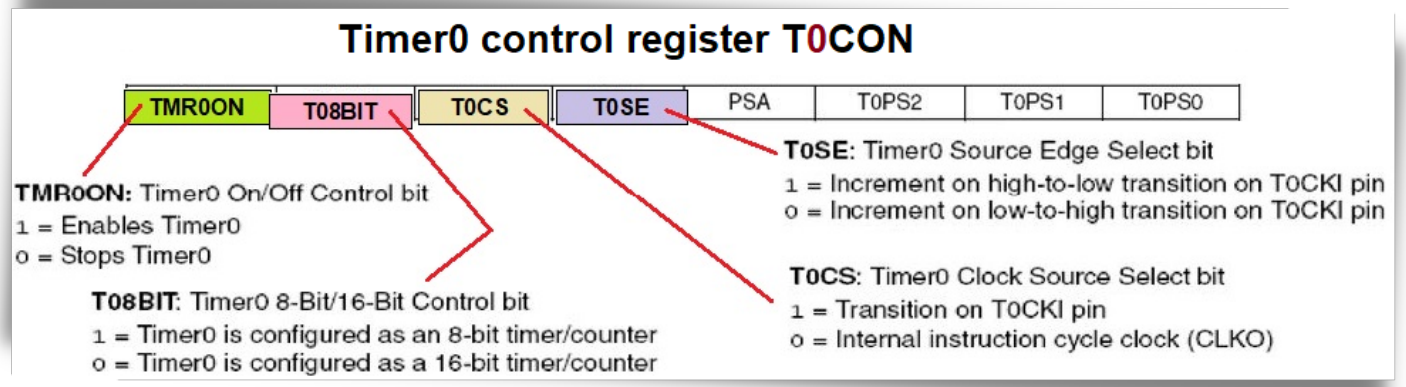
Initial output condition

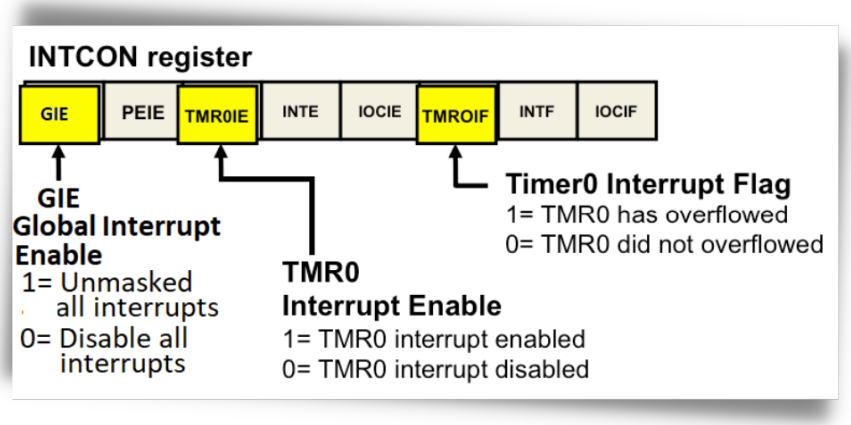
Check if interrupt occur after 10 counts

Output condition after 10 counts

Stop Timer 0.
Clear Timer 0 Interrupt flag.

6. REGISTER
EXPLORE DATA SHEET



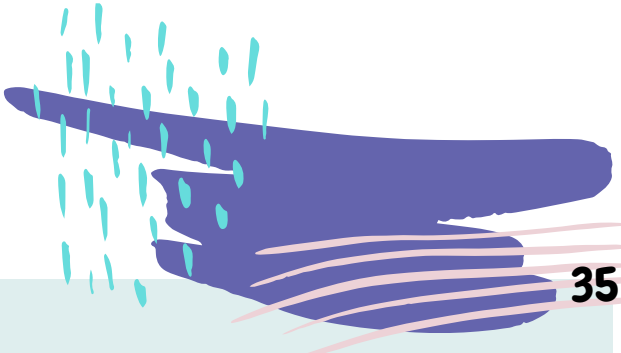


7. ENHANCE UNDERSTANDING

Now, with the same situation, we will change the control system to the following situation.

- A counter sensor will be connected to Pin RA4
- Conveyer motor will be connected to Pin RC5
- Buzzer will be connected to Pin RB3
- Motor will stop moving and buzzer will be switch on when the counter sensor count to 20 bottles fall into the box.
- Use a 16 bit counter system without prescaler.

Rebuild the complete program of the control system.



Date: ----------

My Notes

#BulletJournal

“You deserve to feel like you always matter. Because you always do”

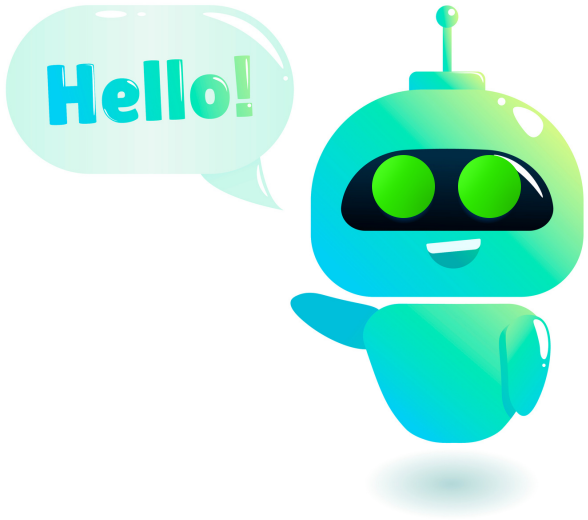
03

project 03

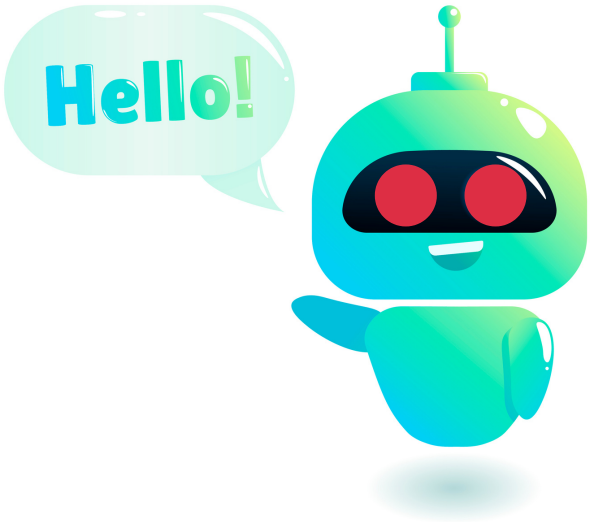
1. PROBLEM STATEMENT

As a programmer, you are required to design a robot toy where a robot eye's will appear red at night time. However during daylight the eye will appear green.
(Tips: Use any PORT in PIC18.

During daylight

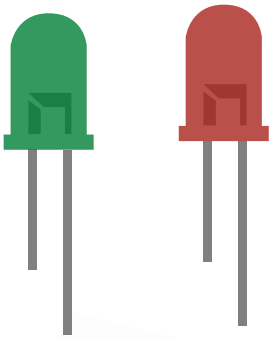


At night



2. CIRCUIT DESIGN

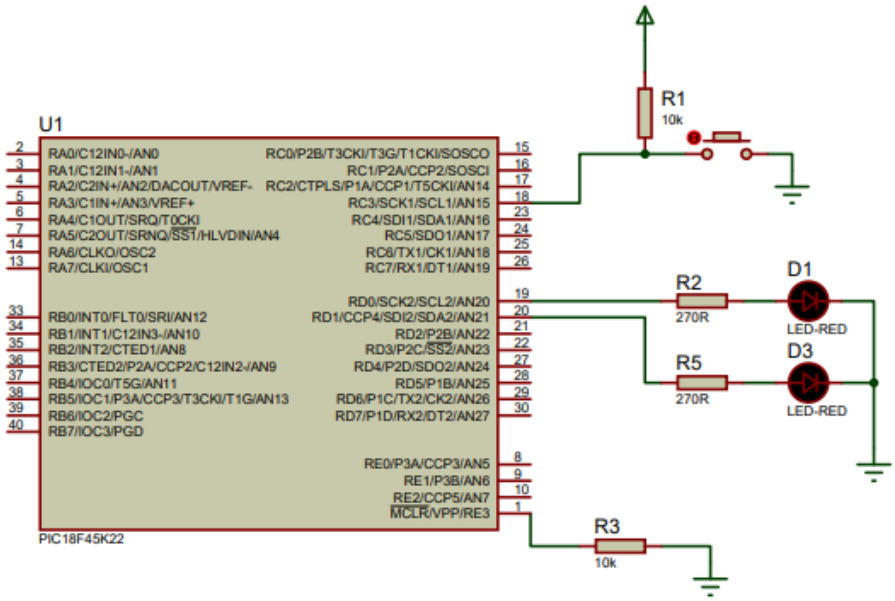
MAIN COMPONENTS

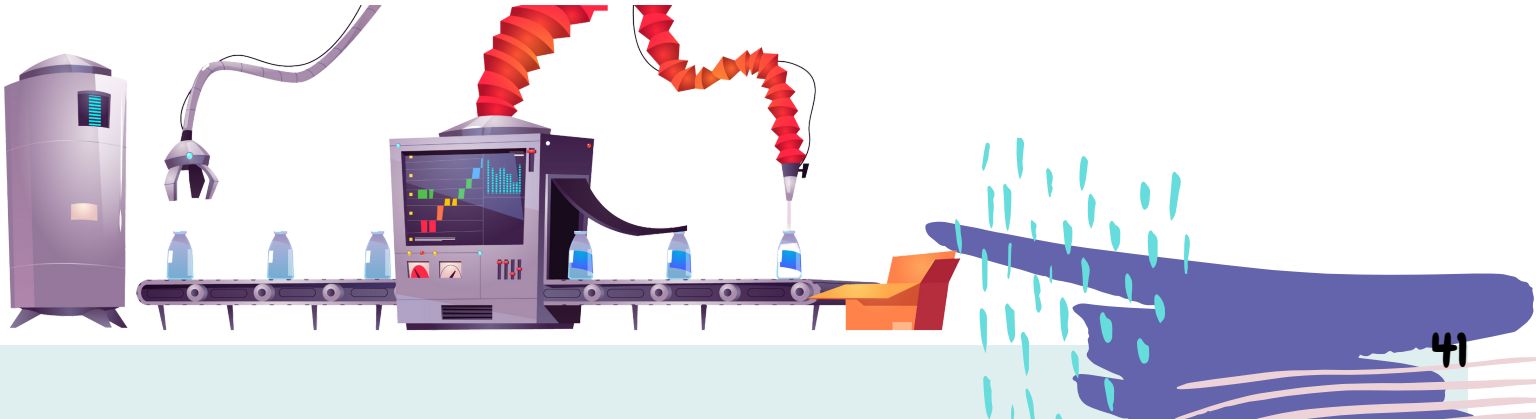


- [x] PIC 18 F45K22
- [x] LDR Sensor
- [x] 1 LED Red
- [x] 1 LED Green
- [x] 2 Resistor

2. CIRCUITDESIGN

PROTEUS SIMULATION





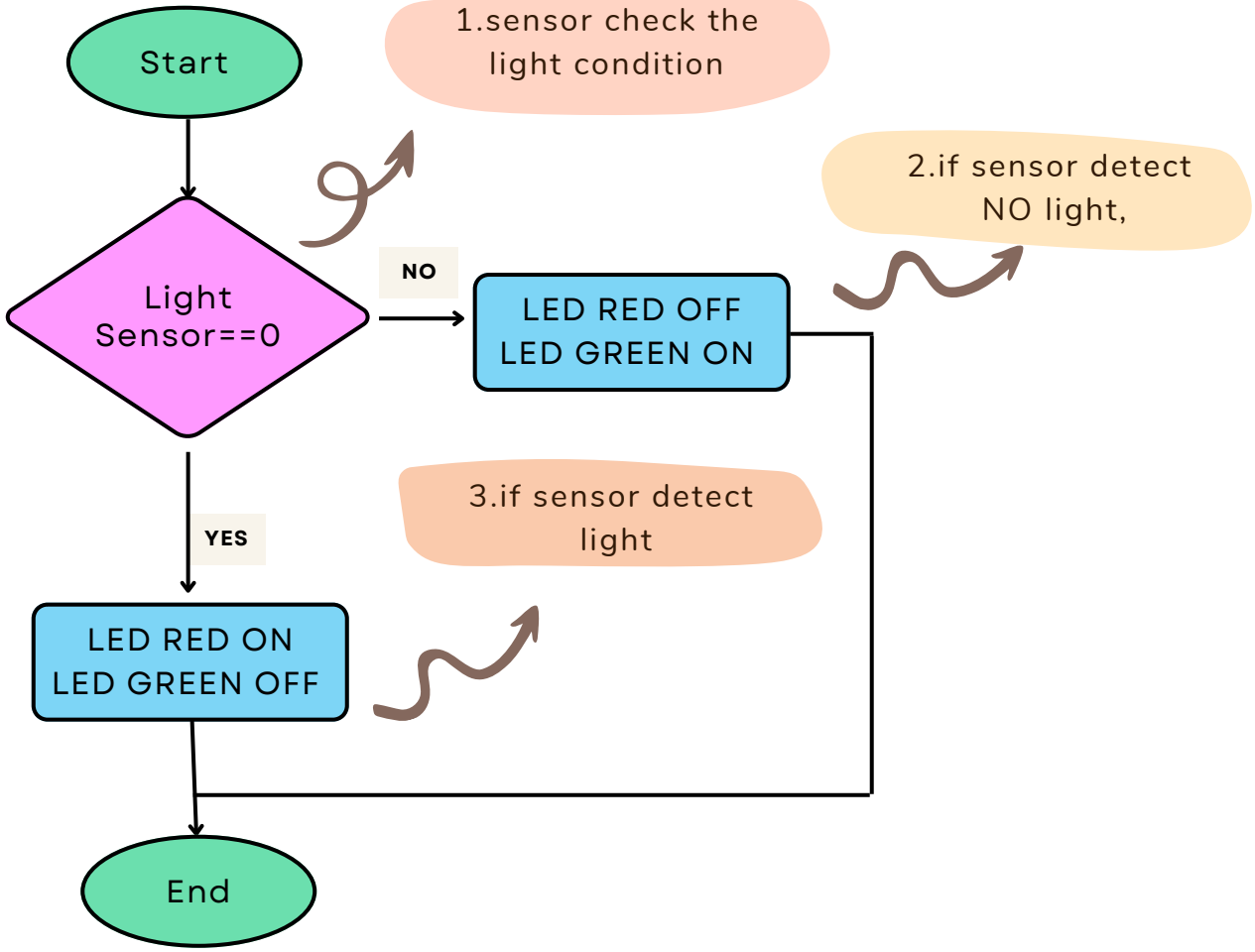
3. CIRCUIT DESIGN
FLOW PROCESS

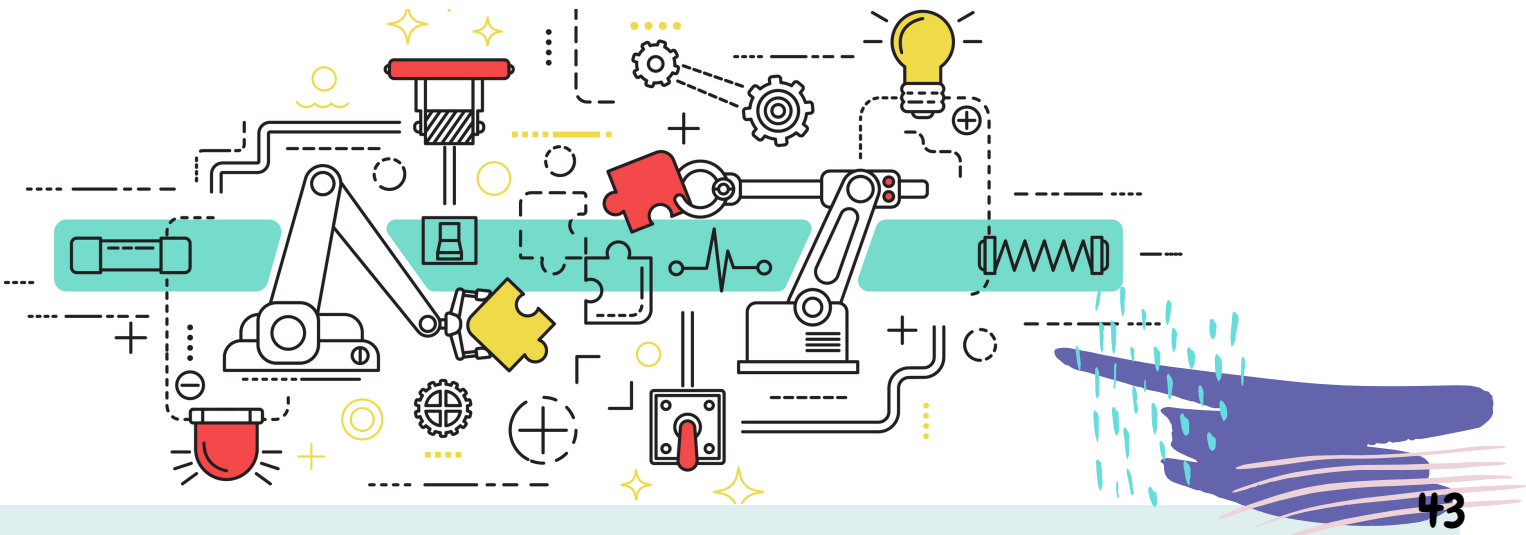
Sensor detect light or no light

if sensor detect NO light, LED RED will blinking

otherwise , LED GREEN will blinking

3.UNDERSTAND
FLOW CHART





4. CONFIGURE
INPUT/OUTPUT

01 Input Device:
1 .Light Sensor

Declare Light Sensor (LDR) as Input
TRISCbits.TRISC3=1;

02 Output Device:
1. LED

Declare LED as Output
TRISD=0;

DID YOU KNOW ?

We can configure 'TRIS' either using Byte or Bit Addressable format

Using Byte Addressable

TRISC=0xFF

If Using Bit Addressable

TRISCbits.TRISC3=1 ;

5. PROGRAMMING
C LANGUAGE

```
#include <xc.h>
#define _XTAL_64000000L
#define LED_GREEN LATDbits.LATD0
#define LED_RED LATDbits.LATD1
#define LIGHT_SENSOR PORTCbits.RC3

void main (void)

{
  TRISCbits.TRISC3=1;
  TRISD=0X00;
  LATD=0;

  while(1)

  {
      if(LIGHT_SENSOR==0)
      {
          LED_GREEN=0;
          LED_RED=1;
      }
      else
      {
          LED_GREEN=1;
          LED_RED=0;
      }
  }
  return;
}
```

Declare PORT for Sensor, LED RED and LED GREEN

Declare each PORT as INPUT/OUTPUT.

Check Sensor Status :

LED RED TURN ON
LED GREEN TURN OFF

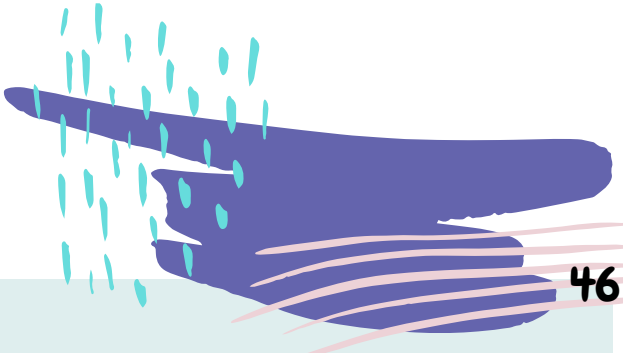
LED RED TURN OFF
LED GREEN TURN ON

7. ENHANCE UNDERSTANDING

Now, with the same situation, we will change the control system to the following situation.

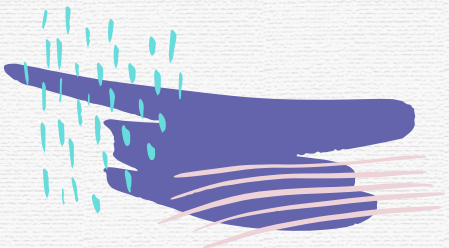
- A counter sensor will be connected to Pin RA4
- Conveyer motor will be connected to Pin RC5
- Buzzer will be connected to Pin RB3
- Motor will stop moving and buzzer will be switch on when the counter sensor count to 20 bottles fall into the box.
- Use a 16 bit counter system without prescaler.

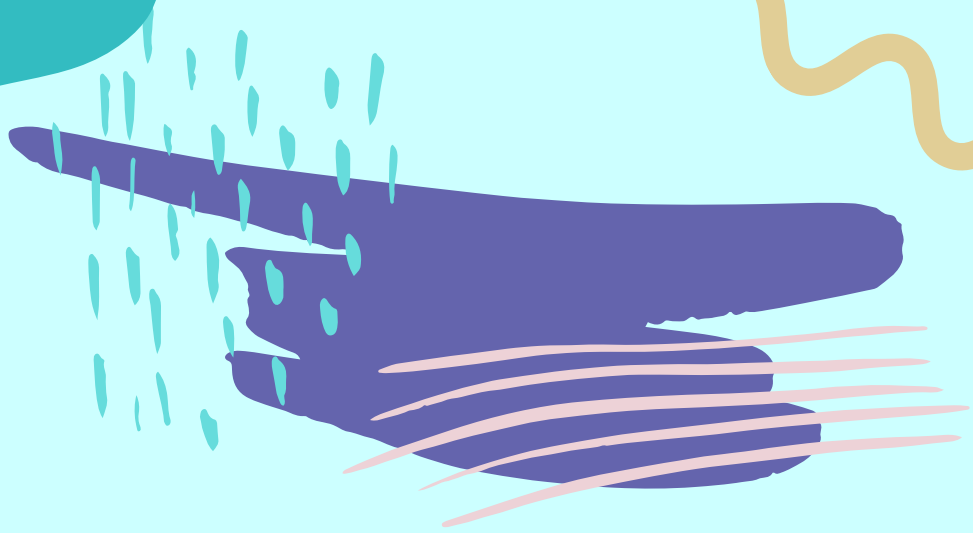
Rebuild the complete program of the control system.



NOTE

/ /





One step at a time. You'll get there.

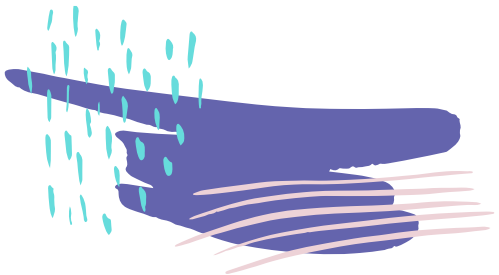
04

project 04

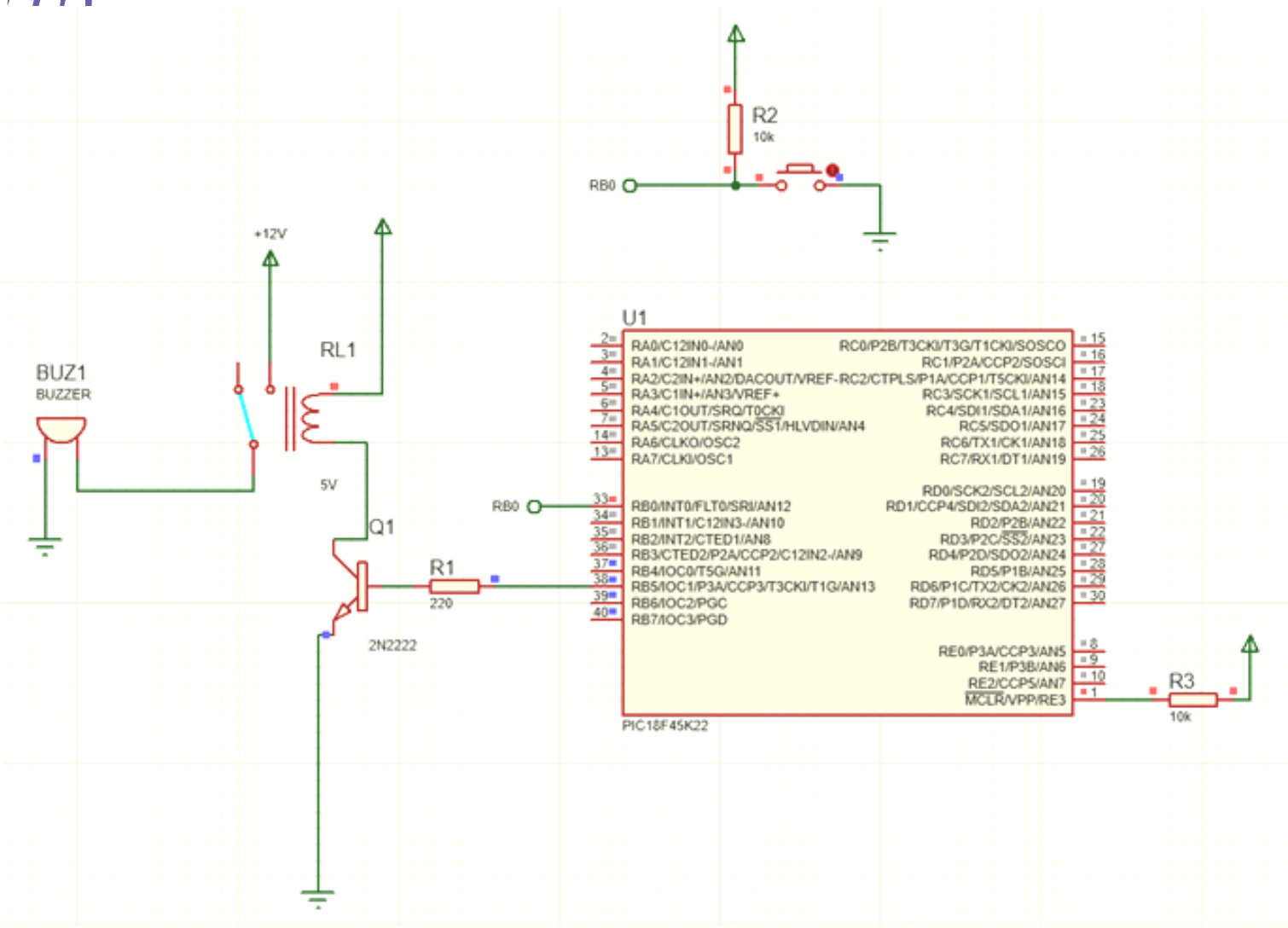
PROBLEM STATEMENT

A security door is attached to a sensor switch (active LOW) which is connected to INT0 and a busser (active HIGH) is connected to RD7. Design a control system so that every time the door is opened, the buzzer beeps twice. Apply interupt programming for this system. Set the interrupt to be triggered on rising edge.

DO IT YOUR WAY



DESIGN CIRCUIT
INPUT/OUTPUT



PROGRAMMING
C LANGUAGE

```c
#include <xc.h>
#define _XTAL_FREQ 20000000

void main(void)
{
    TRISB = 0b00001111;
    INTCON2bits.INTEDG0 = 0;
    INTCONbits.INT0E = 1;
    INTCONbits.INT0F = 0;
    INTCONbits.GIE = 1;

    while (1)
    {

    }
}

void interrupt ISR()
{
    if (INTCONbits.INT0F ==1)
    {
            LATBbits.LATB5 =1;
            __delay_ms(15);
            LATBbits.LATB5 =0;
            __delay_ms(15);
            LATBbits.LATB5 =1;
            __delay_ms(15);
            LATBbits.LATB5 =0;
            __delay_ms(15);
            INTCONbits.INT0F = 0;
    }
```

Check your datasheet to configure the INTERRUPT

this is how we code INTERRUPT

check if INT0 occurs

reset IF, set IF to 0

"Stop making plans, go ahead."

Bob Sadino

05

project 05

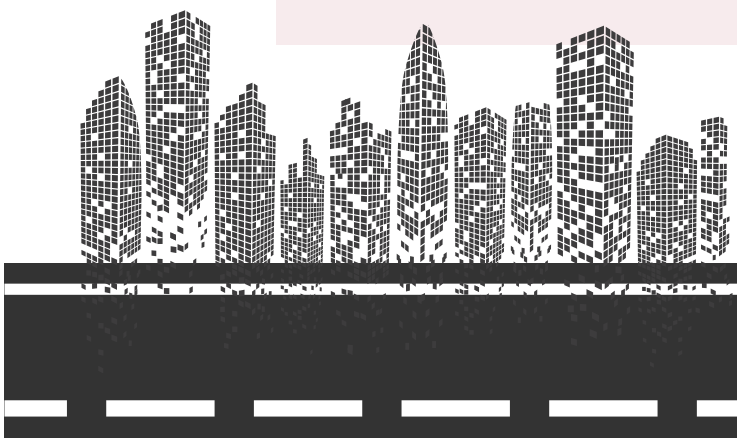
PROBLEM STATEMENT

JV Company Sdn Bhd has been assigned by Majlis Perbandaran Kulim Hitech to design a smart traffic light systems at the busiest traffic light in Kulim Hitech Industrial Area. As a programmer, you are responsible to design the system for pedestrian to cross the road. The specifications are give as below:

(a) Use active LOW push button, an active HIGH buzzer , active HIGH LED for vehicles (RED, YELLOW and GREEN) and active HIGH LED for pedestrianc (GREEN and RED)

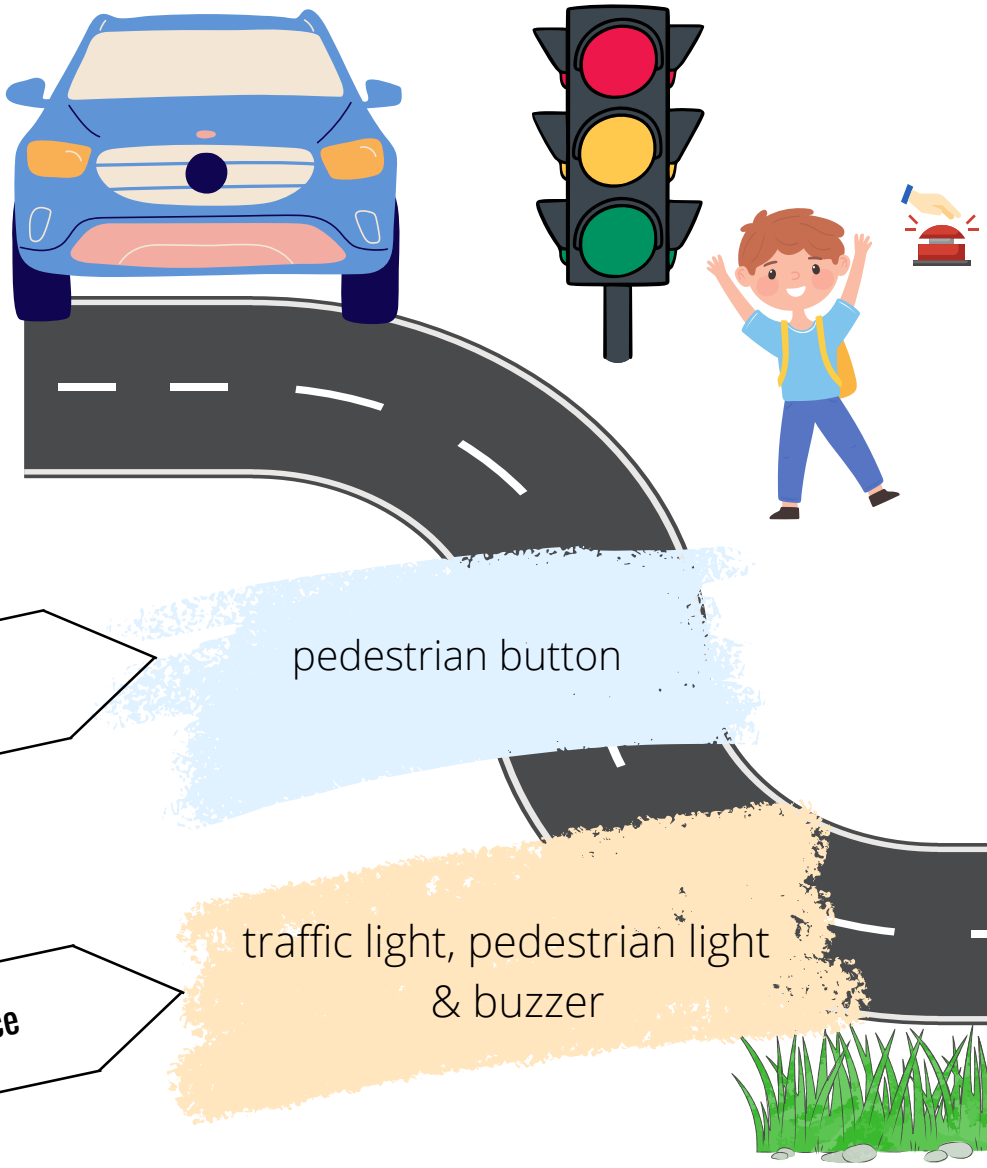
(b) Push button must be connected to RB1/INT1 while LED and buzzer are connected to PORTD

(c) To cross the road, pedestrian has to press the push button that is programmed as external hardware interrupt. The systems will function as normal after the interrupt.



DO IT YOUR WAY

CONFIGURATION
INPUT/OUTPUT



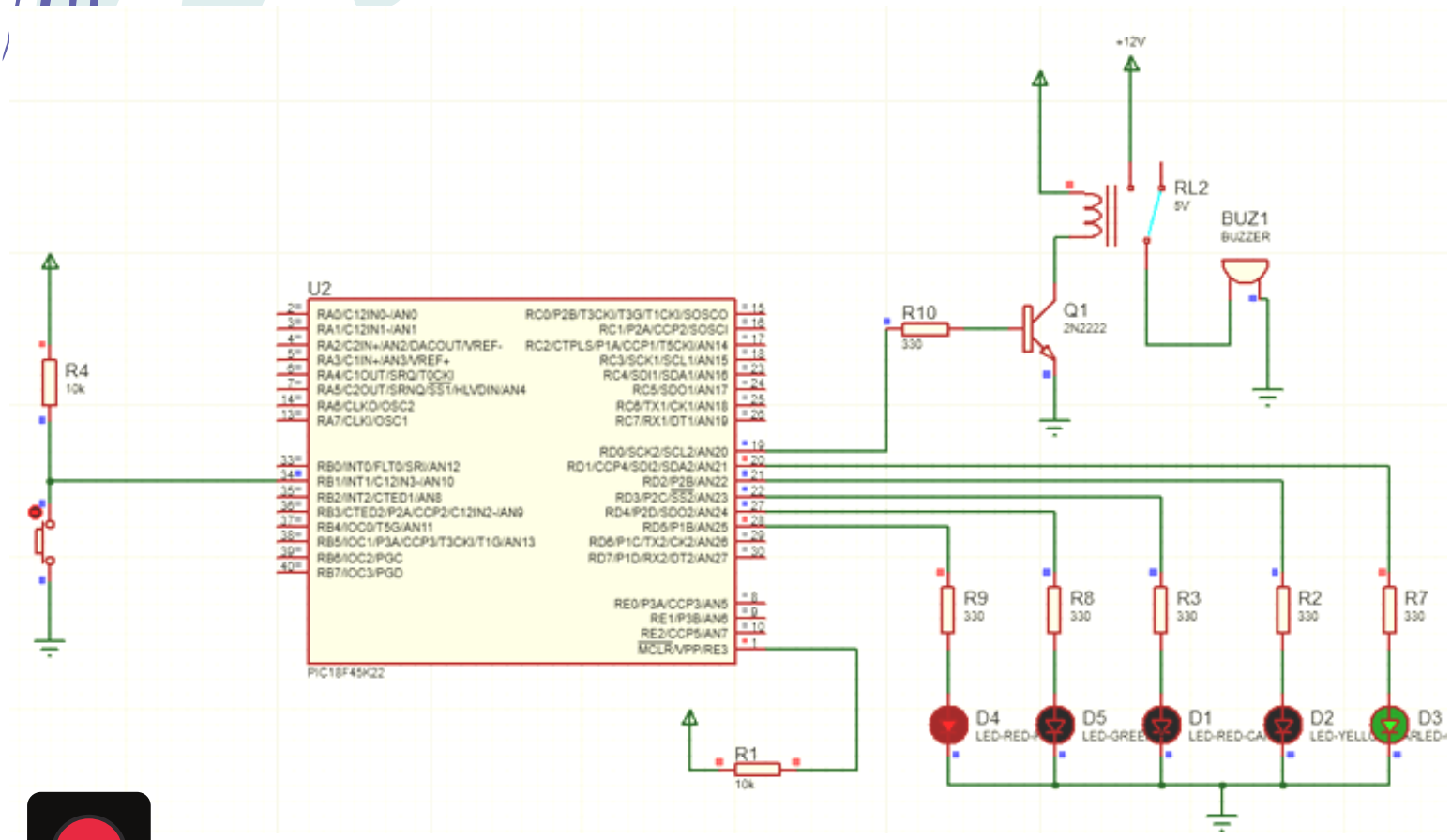
01 **Input Device**

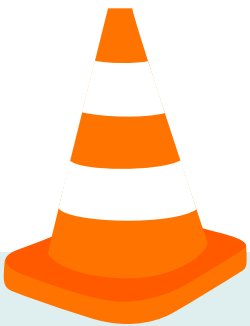
pedestrian button

02 **Output Device**

traffic light, pedestrian light & buzzer

DESIGN CIRCUIT
INPUT/OUTPUT



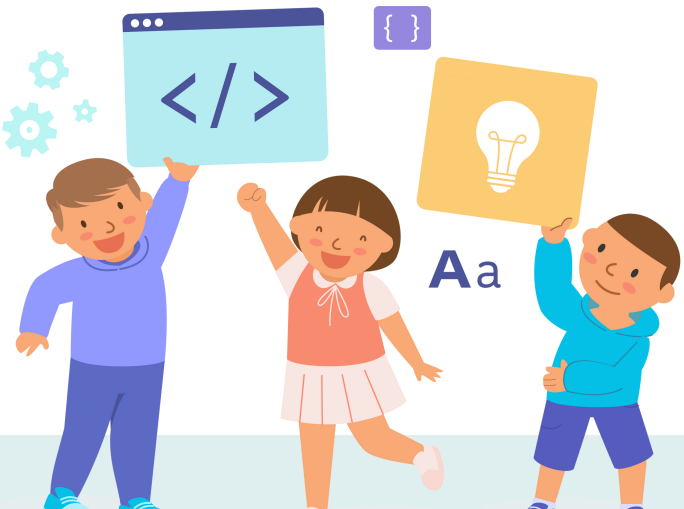


PROGRAMMING
C LANGUAGE

```
#include <xc.h>
#define _XTAL_FREQ 64000000L
#define BZ LATDbits.LATD0
#define LED_G_CAR LATDbits.LATD1
#define LED_Y_CAR LATDbits.LATD2
#define LED_R_CAR LATDbits.LATD3
#define LED_G_P LATDbits.LATD4
#define LED_R_P LATDbits.LATD5


void main(void)
{
    //Your program start here
    TRISDbits.TRISD0 = 0;
    TRISDbits.TRISD1 = 0;
    TRISDbits.TRISD2 = 0;
    TRISDbits.TRISD3 = 0;
    TRISDbits.TRISD4 = 0;
    TRISDbits.TRISD5 = 0;
    TRISBbits.TRISB1 = 1;
    INTCONbits.GIE = 1;
```

Define simple name to all output pin

You also can use byte addressable format. TRISD=0



```
28          INTCON3bits.INT1IE = 1;
29          INTCON3bits.INT1IF = 0;
30          INTCON2bits.INTEDG1 = 0;
31
32
33          while (1)
34          {
35              LED_G_CAR = 1;
36              LED_Y_CAR = 0;
37              LED_R_CAR = 0;
38              LED_G_P = 0;
39              LED_R_P = 1;
40
41          }
42      }
43
44      static void interrupt ISR (void)
45      {
46          if (INTCON3bits.INT1IF == 1)
47          {
48              for(int flash_count = 0; flash_count <1; flash_count ++)
49              {
50              for(int count=0; count<15; count++)
51              LED_G_CAR = 0;
52              LED_Y_CAR = 1;
53              LED_R_CAR = 0;
54              LED_G_P = 0;
55             LED_R_P = 1;
56             __delay_ms (50);
57             for(int count=0; count<10; count++)
58             LED_G_CAR = 0;
59             LED_Y_CAR = 0;
60             LED_R_CAR = 1;
61             LED_G_P = 1;
62             LED_R_P = 0;
63             BZ = 1;
64             __delay_ms (200);
65             for(int count=0; count<10; count++)
66             LED_G_CAR = 1;
67             LED_Y_CAR = 0;
68             LED_R_CAR = 0;
69             LED_G_P = 0;
70             LED_R_P = 1;
71             BZ = 0;
72
73             }
74
75             INTCON3bits.INT1IF = 0;
76          }
77      }
78
```

INTCON3 is set based on design requirements

Initial condition if no intterupt occur.

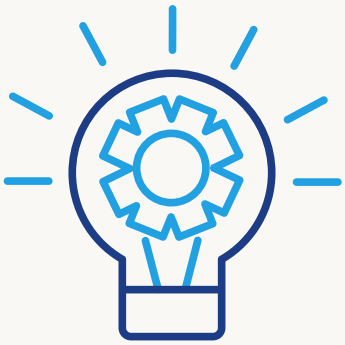
Interrupt occur, 'crossing lane' button is pressed

The Yellow LED is ON, signaling for the car to stop. while Red LED (pedestrian) is still OFF.

The Red LED is ON for the car to stop and pedestrians to cross. The buzzer is also turned ON for 10 counts

After 10 counts, Green LED & pedestrian Buzzer is OFF. The car's Green LED will be turned ON for traffic to return to normal.pin

REFERENCES



Muhammad Ali Mazidi, Danny Causey & Rolin Mc Kinlay (2016). PIC Microcontroller and Embedded Suystems: Using Assembly and C for PIC18. Micro Digital Ed.

PIC18F45K22 Datasheet (PDF) - Microchip Technology," PIC18F45K22 Datasheet (PDF) . [Online]. Available: https://pdf1.alldatasheet.com/datasheet-pdf/view/348759/MICROCHIP/PIC18F45K22.html. [Accessed: 2021].

POLITEKNIK
MALAYSIA
TUANKU SULTANAH BAHIYAH

e ISBN 978-967-2740-11-7

9 7 8 9 6 7 2 7 4 0 1 1 7

PIC18 Based Projects Collection