6 Digital Inputs/Outputs

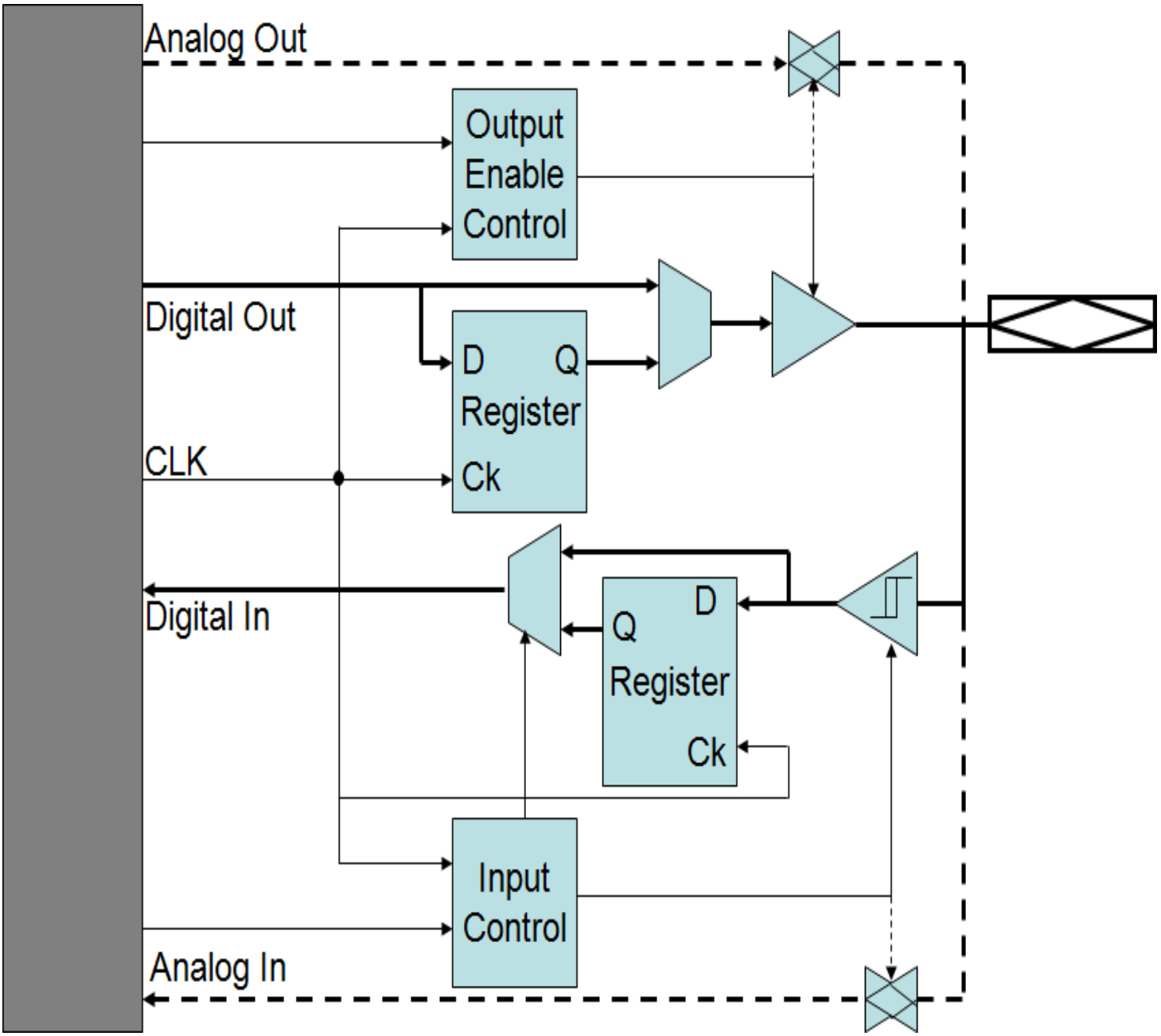


Figure 1 I/O functional pin configuration

1. Objective

Learn how to solve a combinatorial equation, a combinatorial system, and evaluate the response-time of a system, then program a sequential system as states machine.

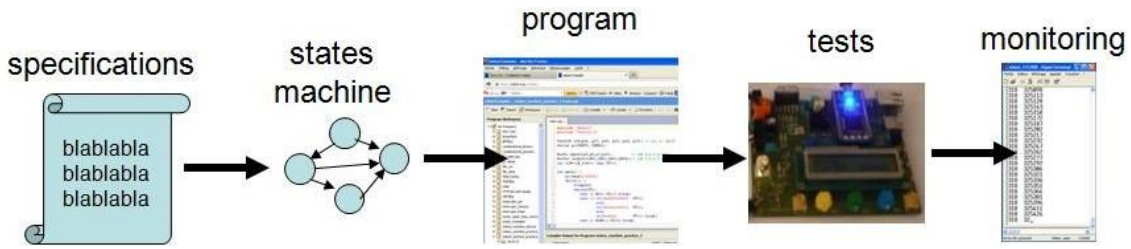
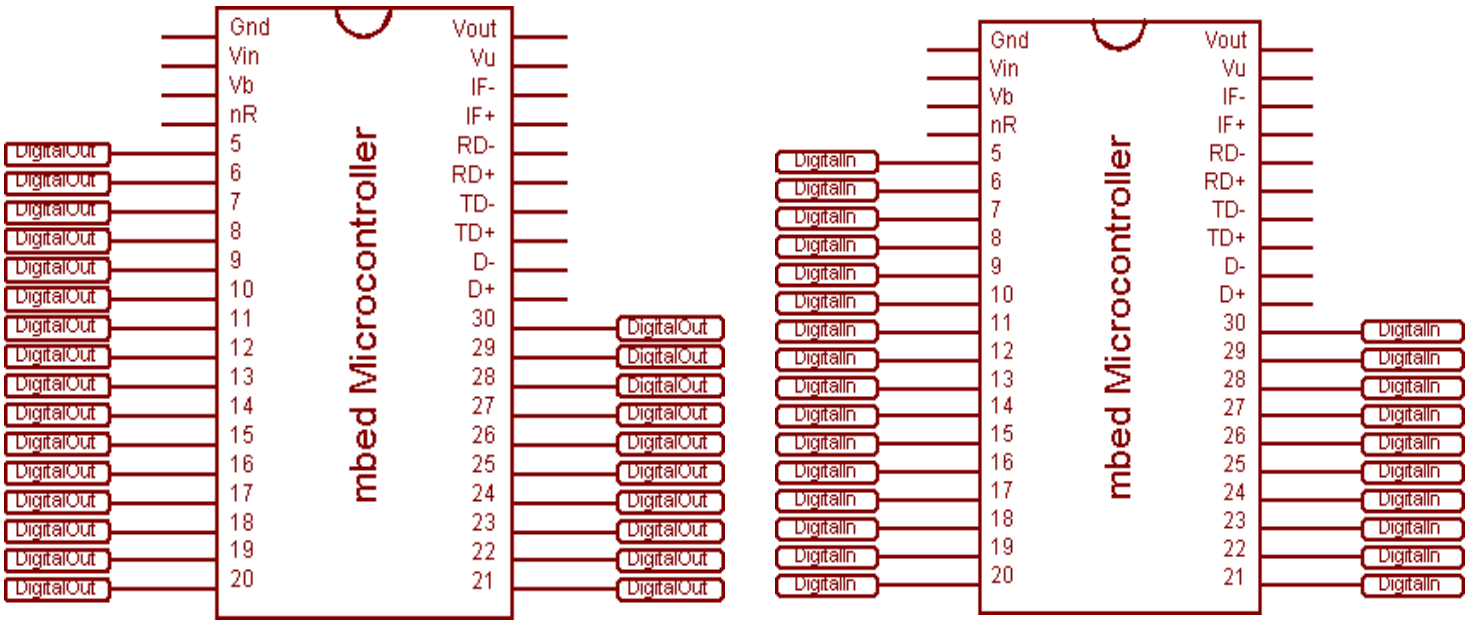


Figure 2 Objective

2. Overview



2.1 Input Pin

	DigitalIn (PinName pin, const char *name=NULL)
	Create a **DigitalIn** connected to the specified pin.
int	**read** ()
	Read the input, represented as 0 or 1 (int)
void	**mode** (PinMode pull)
	Set the input pin mode.
	operator int ()
	An operator shorthand for **read()**

Example code:

```
1. DigitalIn input1 (p5);
2. a = input1.read(); //or a=input1
```

LSB MSB

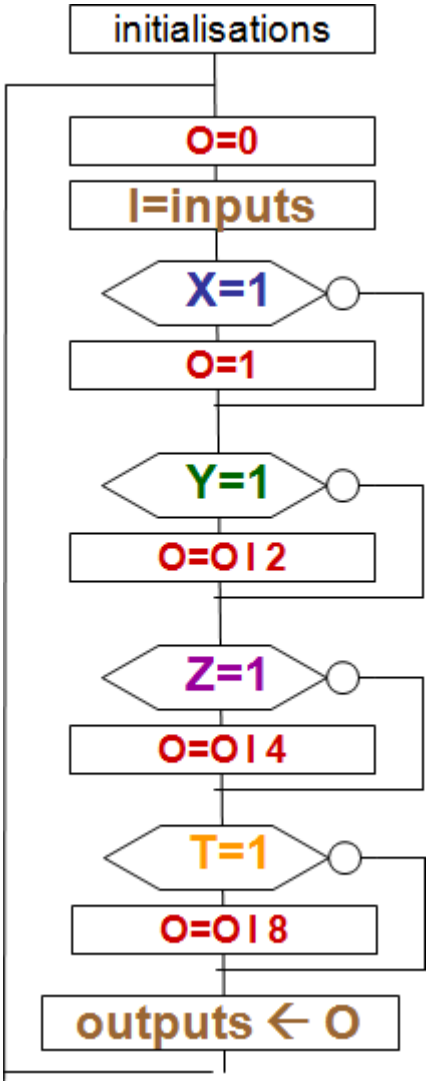
2.2 Input bus

	BusIn (PinName p0, PinName p1=NC, , PinName p15=NC, const char *name=NULL)
	Create an **BusIn** , connected to the specified pins.
int	**read** ()
	Read the value of the input bus.
	operator int ()
	A shorthand for **read()**

Example codes.

```
1. BusIn inputs(p5, p6, p11, p23);
2. a=inputs.read();
3. if(inputs & 0x07==0x05)…..;
```

2.3 Combinatorial system "*Successive evaluation method*"



Sequence diagram

X= /a + b.c + a.b./d
Y= a.b.c + a./d
Z= /b./c./d + a.b./c./d
T= a./c.d + b.d

Outputs equation

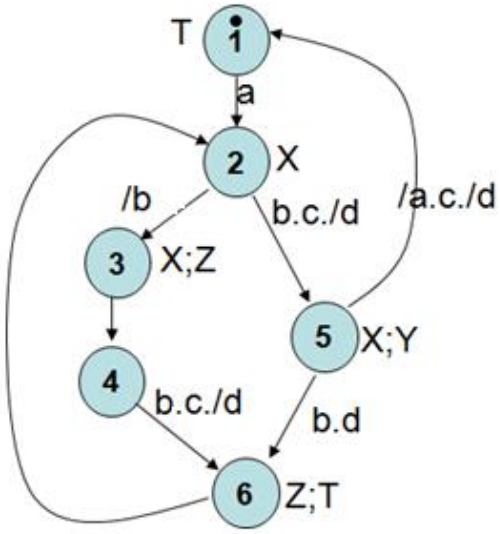
a → p5	LED1 → X
b → p6	LED2 → Y
c → p7	LED3 → Z
d → P16	LED4 → T

Input output mapping

```
#include "mbed.h"
BusIn inputs(p5,p6,p7,p16);
BusOut outputs(LED1,LED2,LED3,LED4);
DigitalIn a(p5),b(p6),c(p7),d(p16);
void main(){
  int I,O;
  while(1){
    a.mode(PullUp);
    b.mode(PullUp);
    c.mode(PullUp);
    d.mode(PullUp);
    I=inputs;O=0;
    if(((I&1)==0) || ((I&6)==6) || ((I&11)==3)) O=1 ;
    if(((I&7)==7) || ((I&9)==1)) O=O|2 ;
    if(((I&14)==0) || ((I&15)==3)) O=O|4 ;
    if(((I&13)==9) || ((I&10)==10)) O=O|8 ;
    outputs=O;
  }
}
```

Program

1.3 Sequential system “States Machine”



FSM

a → p5
b → p6
c → p7
d → P16

LED1 → X
LED2 → Y
LED3 → Z
LED4 → T

Input output mapping

```
#include "mbed.h"
BusIn inputs(p5,p6,p7,p16);
BusOut outputs(LED1,LED2,LED3,LED4);
DigitalIn a(p5),b(p6),c(p7),d(p16);
int I; char ST=1;
int main() {
  a.mode(PullUp);
  b.mode(PullUp);
  c.mode(PullUp);
  d.mode(PullUp);
  while(1) {
      I=inputs;
      switch(ST)
      {
         case 1: outputs=0x8;
                 if ((I & 0x1) == 0x1) ST=2; break;
         case 2: outputs=0x1;
                 if ((I & 0x2) == 0x0) ST=3;
                 else
                 if ((I & 0xe) == 0x6) ST=5; break;
         case 3: outputs=0x5;
                 if (((I & 0x2) == 0x2) || ((I & 0x0) == 0x0))ST=4; break;
         case 4: outputs=0x0;
                   Write code ...
         case 5: outputs-0x3;
                   Write code ...
         case 6: outputs=0xc;
                   Write code ...


       }
  }
}
```

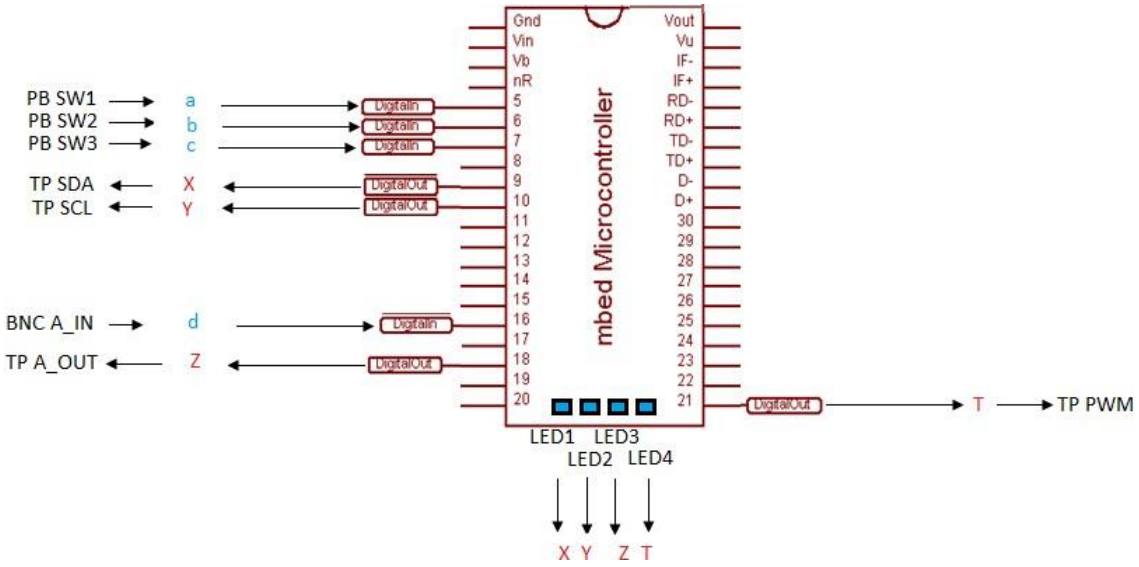
Program

3. Procedure

The mbed module is supplied by the USB interface (no need of external power supply). The supply jumper has to be in the position “USB”.

We will use combinatorial variables a, b, c, d, X, Y, Z and T connected to the following table:

	INPUTS				OUTPUTS			
Name	a	b	c	d	X	Y	Z	T
MBED Pin	P5	P6	P7	P16	LED1	LED2	LED3	LED4
Board	PB SW1	PB SW2	PB SW3	BNC A_IN	TP SDA	TP SCL	TP A_OUT	TP PWM



Remark: there are no external pull up resistors in inputs p5, p6, p7, and p16.

1. **Combinatorial equation**

Using the DigitalIn and DigitalOut classes, write the program solving the equation:

X = /a.b.c + a./b./c.d + a.c.d

- Test the program using the push buttons and watching the LED1.
- Connect a square wave function generator to BNC input corresponding to variable d. IF the buttons are not pushed, a=b=c=1 and X=d. Observe simultaneously d and X with an oscilloscope. Measure the delay between the rising edges of d and X.

2. Combinational system

Using BusIn and BusOut classes, and the successive evaluation method, write the program solving the system:

X = /a.b.c + a./b./c

Y = a.b.c.d + /a./b./c.d

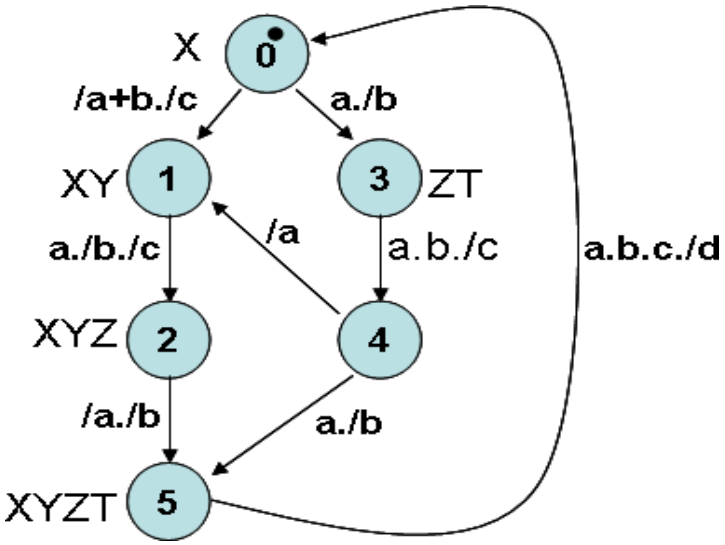
Z = a./c

T = /b.d + c./d

Test the program using the push buttons and watching the LEDs.

3. Sequential system: state machine

3.1 Write and test the program corresponding to the following specifications:



3.2 We will display on the LCD the number of active state.