LỜI NÓI ĐẦU

Ngày nay , vơi những ứng dụng của khoa học kĩ thuật tiên tiến , thế giới của chúng ta đã và đang ngày một thay đổi , văn minh và hiện đại hơn . sự phát triển của kĩ thuật điện tử đã tạo ra hang loạt những thiết bị với các đặc điểm nổi bật như sự chính xác cao , tốc độ nhanh và gọn nhẹ góp phần cho sựu hoạt động của con người đạt được hiệu quả cao.

Gắn liền sự phát của khoa học điện tử là sự phát triển của các vi xử lí, vi điều khiển, đó là sự ra đời của vi xử lí đa năng như Pentium, Celerong … và trong vi điều khiển cũng có bước nhảy vọt được đánh dấu bởi sự ra đời của các vi điều khiển như PIC, AVR, FPGA… các vi xử lí và vi điều khiển này ngày càng đc sử dụng rộng rãi và phổ biến , đặc biệt chúng có thể làm được những việc vô cũng phức tạp. với những ứng dụng phổ biến trong mọi lĩnh vực của cuộc sống.

Và trong môn học vi sử lí này nhóm em đã quyết định làm đề tai " thiết kế mạch điều khiển thang máy dung vi điều khiển AVR" đó là một đề tài thiết thực và rât gần gũi với cuộc sống. nó phục vụ trực tiếp và đắc lực cho việc xây dựng các tào nhà cao ốc trong thời buổi công nghiệp hiên nay.

Mặc dù đã rất cố gắng thiết kế và làm mạch nhưng do thời gian ngắn và năng lực còn hạn chế nên không thể tránh khỏi sai sót . chúng em rất mong sự góp ý và giúp đỡ của thầy cho đề tài của chúng em được hoàn thiện hơn.

Em xin chân thành cám ơn!

PHẦN I: LÍ THUYẾT CHUNG VỀ HỆ THỐNG ĐIỀU KHIỂN THANG MÁY.

Chương 1. Giới thiệu chung hệ thống thang máy.

1.1 Giới thiệu chung về thang máy.

Thang máy là thiết bị vận tải dùng để chở người và hàng hoá theo

phương thẳng đứng. Nó là một loại hình máy nâng chuyển được sử dụng rộng rãi trong các ngành sản xuất của nền kinh tế quốc dân như trong ngành khai thác hầm mỏ, trong ngành xây dựng, luyện kim, công nghiệp nhẹ... ở những nơi đó thang máy được sử dụng để vận chuyển hàng hoá, sản phẩm, đưa công nhân tới nơi làm việc có độ cao khác nhau... Nó đã thay thế cho sức lực của con người và đã mang lại năng suất cao.

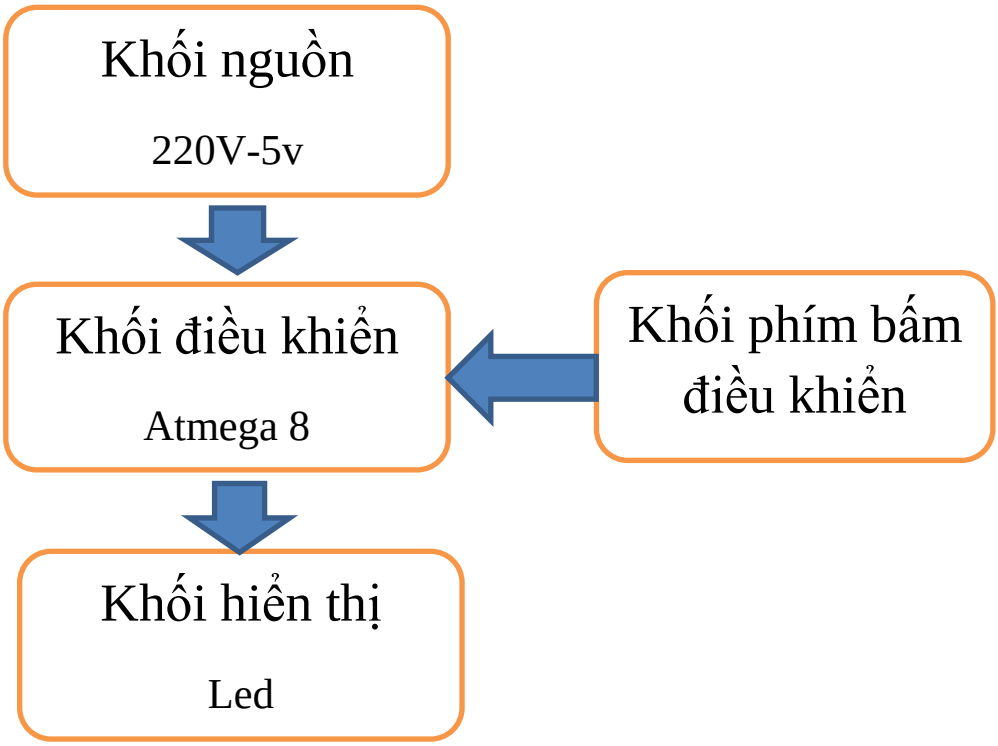
Trong sinh hoạt dân dụng, thang máy được sử dụng rộng rãi trong các toà

nhà cao tầng, cơ quan, khách sạn... Thang máy đã giúp cho con người tiết kiệmđược thời gian và sức lực...

Ở Việt Nam từ trước tới nay thang máy chỉ chủ yếu được sử dụng trong

công nghiệp để trở hàng và ít được phổ biến. Nhưng trong giai đoạn hiện nay nền kinh tế nước ta đang có những bước phát triển mạnh thì nhu cầu sử dụng thang máy trong mọi lĩnh vực ngày càng tăng lên.

1.2. Sơ đồ khối của mạch điều khiển thang máy.



- Giải thích sơ đồ khối của hệ thống

- Khối điều khiển: điều khiển trạng thái hoạt động của thang, là nơi nhận tín hiệu vào và đưa tín hiệu ra cho led 7 thanh.
- Khối phím bấm điều khiển: cung cấp tín hiệu cho khối sử lí trung tâm về hoạt động gọi tầng và chon tầng
- Khối hiển thị: hiển thị vị trí và trạng thái của buồng thang . xác nhận khi có hiện tượng gọi tầng.
- Khối nguồn: cung cấp nguồn 1 chiều cho VDK va led 7 thanh hoạt động.

* Mô tả hoạt động của hệ thống.

1. Khi khởi động cabin ở tầng nào thì ở yên tầng đó. Mặt số sẽ hiện thị vị trí của cabin (tầng số bao nhiêu).

2. Hệ thống luôn kiểm tra các y/c gọi thang theo chiều lên và chiều xuống. Kiểm tra các y/c lên mà không có thì chuyển sang kiểm tra các y/c xuống, y/c xuống mà không có thì lại chuyển sang kiểm tra y/c lên. Hệ thống cứ hoạt động liên tục như

thế. Khi có y/c thì hệ thống sẽ phục vụ y/c, y/c nào gọi trước sẽ đc phục vụ trước, y/c nào gọi sau sẽ đc phục vụ sau. Bất cứ một y/c gọi nào cũng được hệ thống nhớ lại và khi thực hiện xong y/c nào thì xóa y/c ấy đi. Nếu có nhiều y/c gọi thì hệ thống sẽ xử lý ưu tiên theo thứ tự như được trình bày ở phần dưới.

3. Khi cabin đã nhận một y/c gọi lên và đang thực hiện y/c này (cabin đang chạy theo chiều lên) thì các y/c gọi lên từ vị trí cabin trở lên sẽ đc phục vụ trên đường đi của nó (y/c là gồm có y/cgọi thang của khách ở ngoài cabin và y/c chọn tầng đến của khách ở trong cabin), các y/c gọi lên từ vị trí cabin trở xuống sẽ bị bỏ qua. Các y/c gọi xuống cũng sẽ bị bỏ qua nếu tầng trên nó còn có y/c (bỏ qua ở đây là không được phục ngay mà trạng thái gọi sẽ được nhớ lại để

phục vụ sau).

5. Bộ phận hiển thị sẽ hiển thị vị trí của cabin, chiều chạy (là lên hoặc xuống) và cabin đangchạy hay đang dừng.

Chương 2: Yêu cầu chức năng của hệ thống.

2.1. Yêu cầu chức năng:

- Mạch thực hiện đúng yêu cầu mong muốn

- Hiển thị số tầng lên LED 7 thanh

Sử dụng nút bấm để điều khiển hoạt động thang máy.

2.2 Yêu cầu phi chức năng.

Sử dụng vi điều khiển AVR atmega8

Thiết kế mạch nhỏ hơn 1dm2

Thời gian thực hiện đề tài 13 tuần

Mục tiêu thiết kế

Sử dụng nút bấm để điều chỉnh tầng

Sử dụng led 7 thanh để hiển thị trạng thái tầng và led đơn hiển thị vị trí tầng.

Kế hoạch thực hiện

Sau khi phân tích hệ thống, và khả năng nhóm phát triển cũng như quá trình vận hành hệ thống. Kế hoạch có thể thực hiện theo trình tự

Tên quá trình	Thời gian	Bắt đầu	Hoàn thành
Nghiên cứu sơ bộ	1 tuần		
Phân tích	3 tuần		
Lập trình, mô phỏng	2 tuần		
Layout	2 tuần		
Đặt mạch in	1.5 tuần		
Hàn mạch in và làm báo cáo	tuần		

PHẦN II: XÂY DỰNG THUẬT TOÁN VÀ THIẾT KẾ MẠCH

Chương 3. Phân tích kiến trúc hệ thống

Hệ thống gồm 4 khối chính

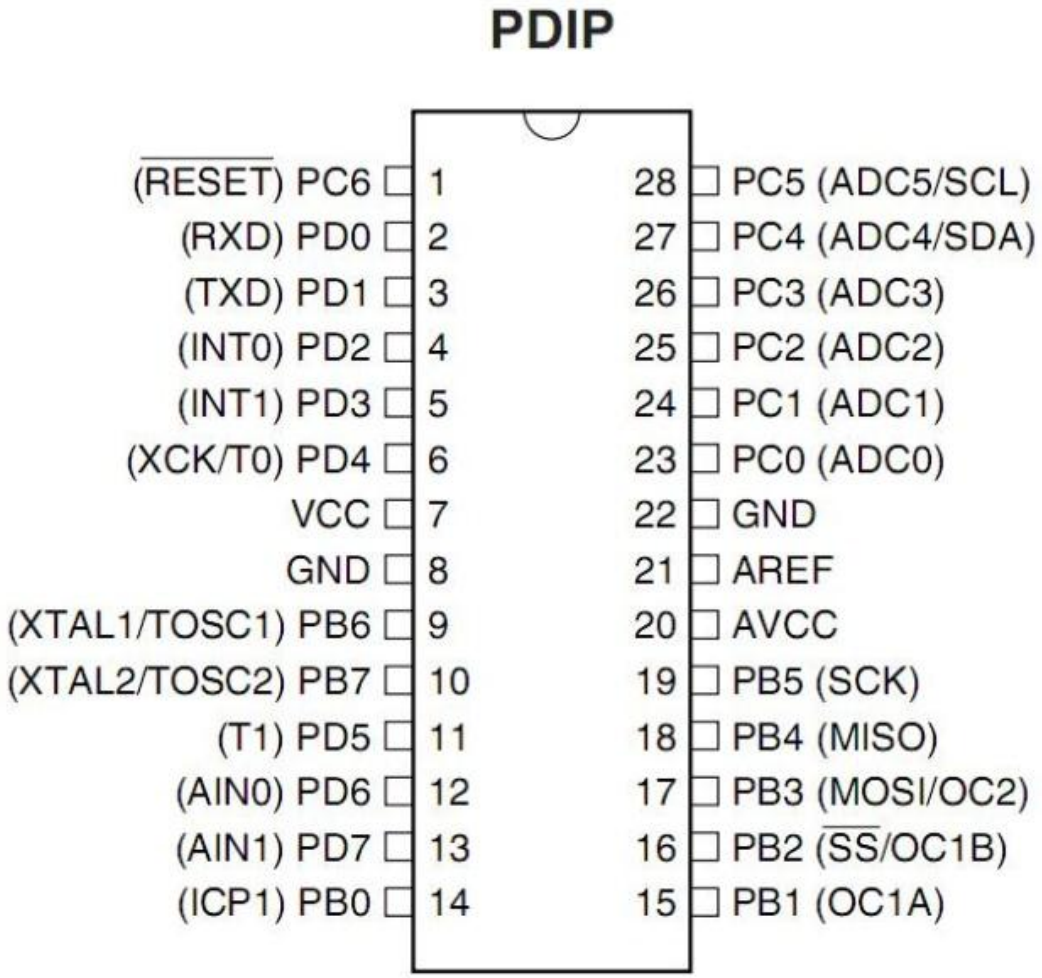
Vi xử lý trung tâm avr ATMEGA8

Khối nguồn

Khối led 7 thanh và led đơn

Khối nút bấm.

3.1. Vi xử lí trung tâm ATMEGA 8



a) Một số đặc tính kỹ thuật

- ATMega8 là một con Vi Điều Khiển thuộc dòng Mega AVR của hãng ATMEL. Dòng Vi Điều Khiển này có tính năng nổi trộ như:
- 32 thanh ghi đa dụng
- Tốc độ tối đa lên đến 16MIPS với thạch anh 16MHz
- Có 8KB bộ nhớ Flash lập trình ISP
  - 512 byte EFPROM.
  - 1k SRAM.
  - Chu kỳ ghi/ xóa 10000 lần cho bộ nhớ flash ROM và 100000 lần cho bộ nhớ EFPROM.
- Tính năng ngoại vi
  - 2 bộ timer/counter 8 bít, 1 bộ so sánh.
  - 1 bộ timer/counter 16 bit.
  - Bô đếm thời gian thực với đao động riêng.
  - 3 kênh PWM.
  - 6 kênh ADC 10 bits cho kiểu vỏ PDIP, 8 kênh ADC 10 bits cho kiểu vỏ TQFP.
  - Giao tiếp nối tiếp TWI, 2 chân ngắt ngoài INT0 và INT1 ứng với 2 chân PD2 và PD3.
  - Lập trình nối tiếp USART, giao tiếp nối tiếp SPI master/slave.
  - Bộ so sánh analog on-chip.
- I/O
  - 23 ngõ vào/ra khả trình.
  - Được đóng gói trong 28 chân kiểu vỏ PDIP.
  - Điện áp hoạt động 2,7V-5,5V(ATmega8L)và 4,5-5,4V(ATmega8).
  - Tần số hoạt động 0-8MHz(ATmega8L) và 0-16MHz(ATmega8) động.

b. Một số chức năng của ATmega8 sử dụng trong bài tập lớn.

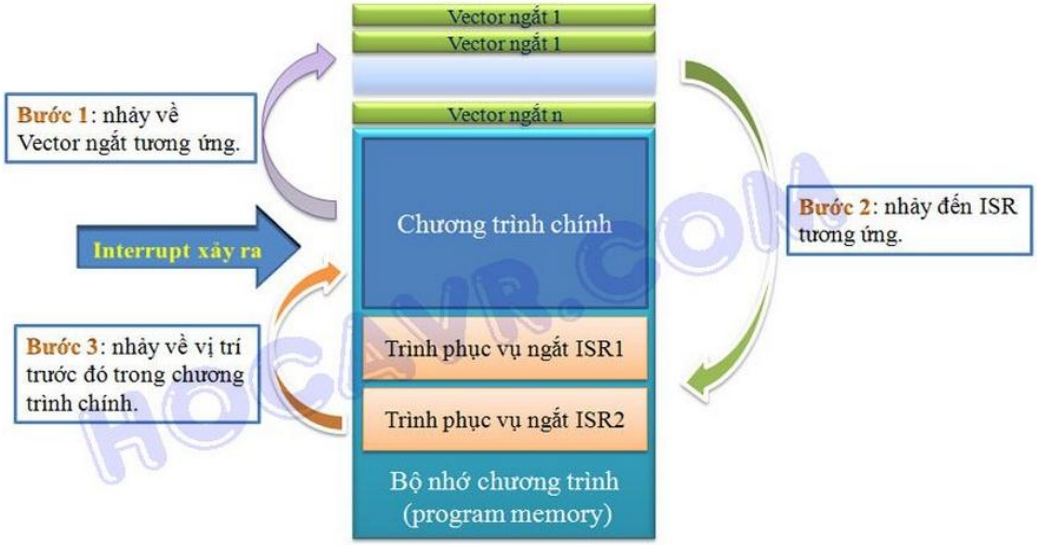
- Ngắt ngoài:

Interrupts, thường được gọi là ngắt, là một tín hiệu khẩn cấp gởi đến bộ xử lí, yêu cầu bộ xử lí tạm ngừng tức khắc các hoạt động hiện tại để "nhảy" đến một nơi khác thực hiện một nhiệm vụ khẩn cấp nào đó,

nhiệm vụ này gọi là trình phục vụ ngắt – isr (interrupt service routine). Sau khi kết thúc nhiệm vụ trong isr, bộ đếm chương trình sẽ được trả về giá trị trước đó để bộ xử lí quay về thực hiện tiếp các nhiệm vụ còn dang dở. Như vậy, ngắt có mức độ ưu tiên xử lí cao nhất, ngắt thường được dùng để xử lí các sự kiện bất ngờ nhưng không tốn quá nhiều thời gian.

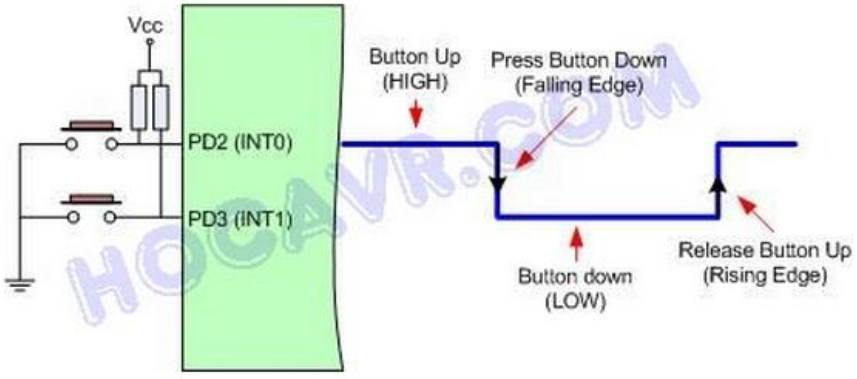
Các tín hiệu dẫn đến ngắt có thể xuất phát từ các thiết bị bên trong chip (ngắt báo bộ đếm timer/counter tràn, ngắt báo quá trình gởi dữ liệu bằng RS232 kết thúc…) hay ***do các tác nhân bên ngoài (ngắt báo có 1 button được nhấn, ngắt báo có 1 gói dữ liệu đã được nhận…).***

Hình minh họa cách tổ chức ngắt thông thường trong các chip AVR:



Có 3 thanh ghi liên quan đến ngắt ngoài đó là MCUCR, GICR và GIFR:

- ***Thanh ghi điều khiển MCU – MCUCR (MCU Control Register) là thanh ghi xác lập chế độ ngắt cho ngắt ngoài:***



Nếu không nhấn, trạng thái các chân INT là HIGH do điện trở kéo lên, khi vừa nhấn 1 button, sẽ có chuyển trạng thái từ HIGH sang LOW, chúng ta gọi là cạnh xuống - **Falling Edge**, khi button được nhấn và giữ, trạng thái các chân INT được xác định là LOW và cuối cùng khi thả các button, trạng thái chuyển từ LOW sang HIGH, gọi là cạnh lên – **Rising Edge**.

Dưới đây là cấu trúc thanh ghi MCUCR được trích ra từ datasheet của chip atmega8.

Bit	7	6	5	4	3	2	1	0	
	SE	SM2	SM1	SM0	ISC11	ISC10	ISC01	ISC00	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

MCUCR là một thanh ghi 8 bit nhưng đối với hoạt động ngắt ngoài, chúng ta chỉ quan tâm đến 4 bit thấp của nó (4 bit cao dùng cho Power manager và Sleep Mode). Bốn bit thấp là các bit Interrupt Sense Control (ISC) trong đó 2 bit ISC11:ISC10 dùng cho INT1 và 2 bit ISC01:ISC00 dùng cho INT0. Hãy nhìn vào bảng tóm tắt bên dưới để biết chức năng của các bit trên, đây là bảng "chân trị" của 2 bit ISC11, ISC10. Bảng chân trị cho các bit ISC01, ISC00 hoàn toàn tương tự.

ISC11	ISC10	Description (Mô tả)
0	0	The low level of INT1 generates an interrupt request. (Mức thấp của chân INT1 tạo ra 1 yêu cầu ngắt – ngắt mức thấp)
0	1	Any logical change on INT1 generates an interrupt request. (Bất kỳ sự thay đổi nào của chân INT1 tạo ra 1 yêu cầu ngắt)
1	0	The falling edge of INT1 generates an interrupt request. (Cạnh xuống trên chân INT1 tạo ra 1 yêu cầu ngắt – ngắt cạnh xuống)
1	1	The rising edge of INT1 generates an interrupt request. (Cạnh lên trên chân INT1 tạo ra 1 yêu cầu ngắt – ngắt cạnh xuống)

- ***Thanh ghi điều khiển ngắt chung – GICR (General Interrupt Control Register).***

  GICR cũng là 1 thanh ghi 8 bit nhưng chỉ có 2 bit cao (bit 6 và bit 7) là được sử dụng cho điều khiển ngắt, cấu trúc thanh ghi như bên dưới (trích datasheet).

Bit	7	6	5	4	3	2	1	0	
	INT1	INT0	–	–	–	–	IVSEL	IVCE	GICR
Read/Write	R/W	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit 7 – INT1 gọi là bit cho phép ngắt 1(Interrupt Enable), set bit này bằng 1 nghĩa bạn cho phép ngắt INT1 hoạt động, tương tự, bit INT0 điều khiển ngắt INT0.

- ***Thanh ghi cờ ngắt chung – GIFR (General Interrupt Flag Register).***

Có 2 bit INTF1 và INTF0 là các bit trạng thái (hay bit cờ - Flag) của 2 ngắt INT1 và INT0. Nếu có 1 sự kiện ngắt phù hợp xảy ra trên chân INT1, bit INTF1 được tự động set bằng 1 (tương tự cho trường hợp của INTF0), chúng ta có thể sử dụng các bit này để nhận ra các ngắt, tuy nhiên điều này là không cần thiết nếu chúng ta cho phép ngắt tự động, vì vậy thanh ghi này thường không được quan tâm khi lập trình ngắt ngoài. Cấu trúc thanh ghi GIFR được trình bày trong hình ngay bên dưới.

Bit	7	6	5	4	3	2	1	0	
	INTF1	INTF0	–	–	–	–	–	–	GIFR
Read/Write	R/W	R/W	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- Giao tiếp TWI-I^2C.

TWI (Two-Wire Serial Intereafce) là một module truyền thông nối tiếp đồng bộ trên các chip AVR dựa trên chuẩn truyền thông I^2C.

TWI trên AVR được vận hành bởi 5 thanh ghi bao gồm thanh ghi tốc độ giữ nhịp TWBR, thanh ghi điều khiển TWCR , thanh ghi trạng thái TWSR, thanh ghi địa chỉ TWAR và thanh ghi dữ liệu TWDR.

- **TWBR (TWI Bit Rate Register)**: là 1 thanh ghi 8 bit quy định tốc độ phát xung giữ nhịp trên đường SCL của chip Master.

7	6	5	4	3	2	1	0	
TWBR7	TWBR6	TWBR5	TWBR4	TWBR3	TWBR2	TWBR1	TWBR0	TWBR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Tốc độ phát xung giữ nhịp được tính theo công thức:

$$\text{SCL frequency} = \frac{\text{CPU Clock frequency}}{16 + 2(\text{TWBR}) \cdot 4^{TWPS}}$$

Trong đó CPU Clock frequency là tần số hoạt động chính của AVR, TWBR là giá trị thanh thi TWBR và TWPS là giá trị của 2 bits TWPS1 và TWPS0 nằm trong thanh thi trạng thái TWSR. Hai bits này được gọi là bit prescaler.

- **TWCR (TWI Control Register)**: là thanh ghi 8 bit điều khiển hoạt động của TWI.

7	6	5	4	3	2	1	0	
TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE	TWCR
R/W	R/W	R/W	R/W	R	R/W	R	R/W	

Một điều cần chú ý là các bit trong thanh ghi TWCR không cần được set cùng lúc, tùy vào từng giai đoạn trong quá trình giao tiếp TWI các bit có thể được set riêng lẻ.

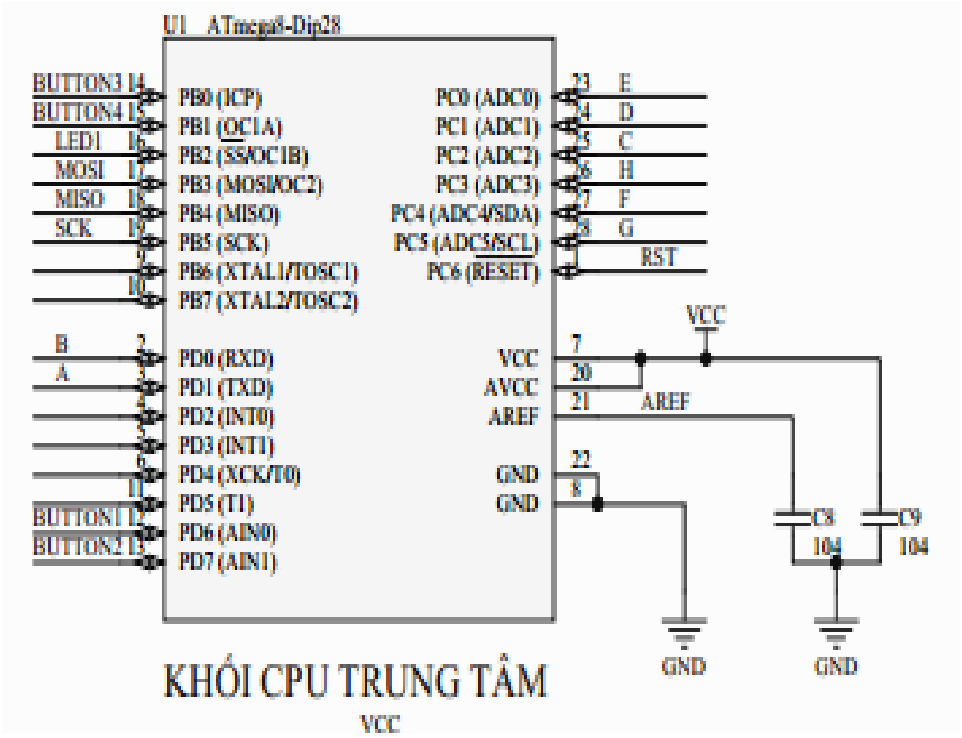
- **TWSR (TWI Status Register)**: là 1 thanh ghi 8 bit trong đó có 5 bit chứa code trạng thái của TWI và 2 bit chọn prescaler.

7	6	5	4	3	2	1	0	
TWS7	TWS6	TWS5	TWS4	TWS3	–	TWPS1	TWPS0	TWSR
R	R	R	R	R	R	R/W	R/W	

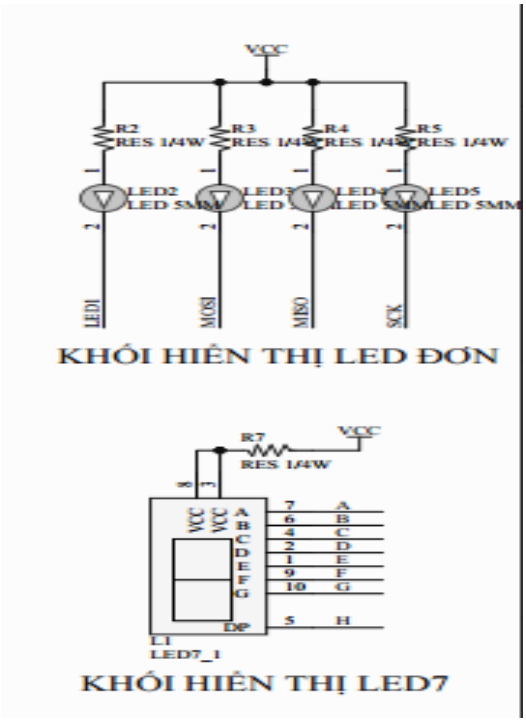
- **TWDR (TWI Data Register)**: là thanh ghi dữ liệu chính của TWI. Trong quá trình nhận, dữ liệu nhận về sẽ được lưu trong TWDR. Trong quá trình gởi, dữ liệu chứa trong TWDR sẽ được chuyển ra đường SDA.

- **TWAR (TWI Address Register)**: là thanh ghi chứa device address của chip Slave. Cấu trúc thanh ghi được trình bày trong hình dưới.

7	6	5	4	3	2	1	0	
TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE	TWAR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	



3.2. Khối LED hiển thị.



LED đơn là một đi-ốt, nó chứa một chíp bán dẫn có pha các tạp chất để tạo ra một tiếp giáp P-N, kênh P chứa lỗ trống, kênh N chứa điện tử, dòng điện truyền từ A-nốt(kênh P) đến K-tốt (kênh N), khi điện tử lấp đầy chỗ trống nó sinh ra bức xạ ánh sáng, các bước sóng phát ra có màu khác nhau tùy thuộc vào tạp chất trong chíp bán dẫn

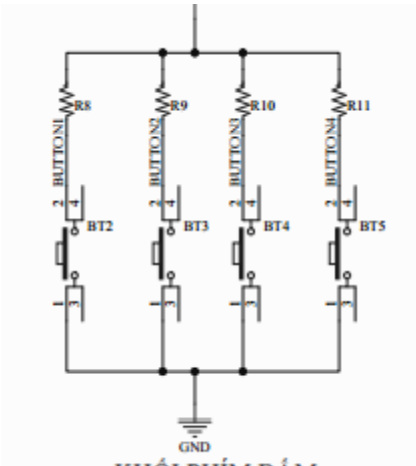
Led 7 đoạn có cấu tạo bao gồm 7 led đơn có dạng thanh xếp theo hình 8 và có thêm một led đơn hình tròn nhỏ thể hiện dấu chấm tròn ở góc dưới, bên phải của led 7 đoạn.

8 led đơn trên led 7 đoạn có Anode(cực +) được nối chung với nhau vào một điểm, được đưa chân ra ngoài để kết nối với mạch điện. 8 cực còn lại trên mỗi led đơn được đưa thành 8 chân riêng, cũng được đưa ra ngoài để kết nối với mạch điện. Đầu chung này được nối với +Vcc, các chân còn lại dùng để điều khiển trạng thái sáng tắt của các led đơn, led chỉ sáng khi tín hiệu đặt vào các chân này ở mức 0.

Vì led 7 đoạn chứa bên trong nó các led đơn, do đó khi kết nối cần đảm bảo dòng qua mỗi led đơn trong khoảng 10mA-20mA để bảo vệ led. Nếu kết nối với nguồn 5V có thể hạn dòng bằng điện trở 330Ω trước các chân nhận tín hiệu điều khiển.

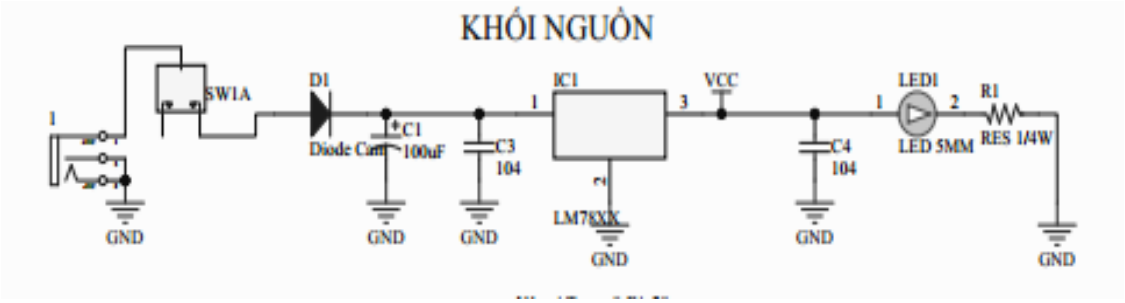
3.3. khối phím bấm

Việc sử dụng phím bấm ở đây dựa trên sự chênh áp khi ấn các phím, điện áp được đưa về chip để xử lí . như vậy việc dựa vào giá trị điện áp đưa vào chân chúng ta có thể hpanf toàn nhận biết được chúng ta đang bấm phím nào?



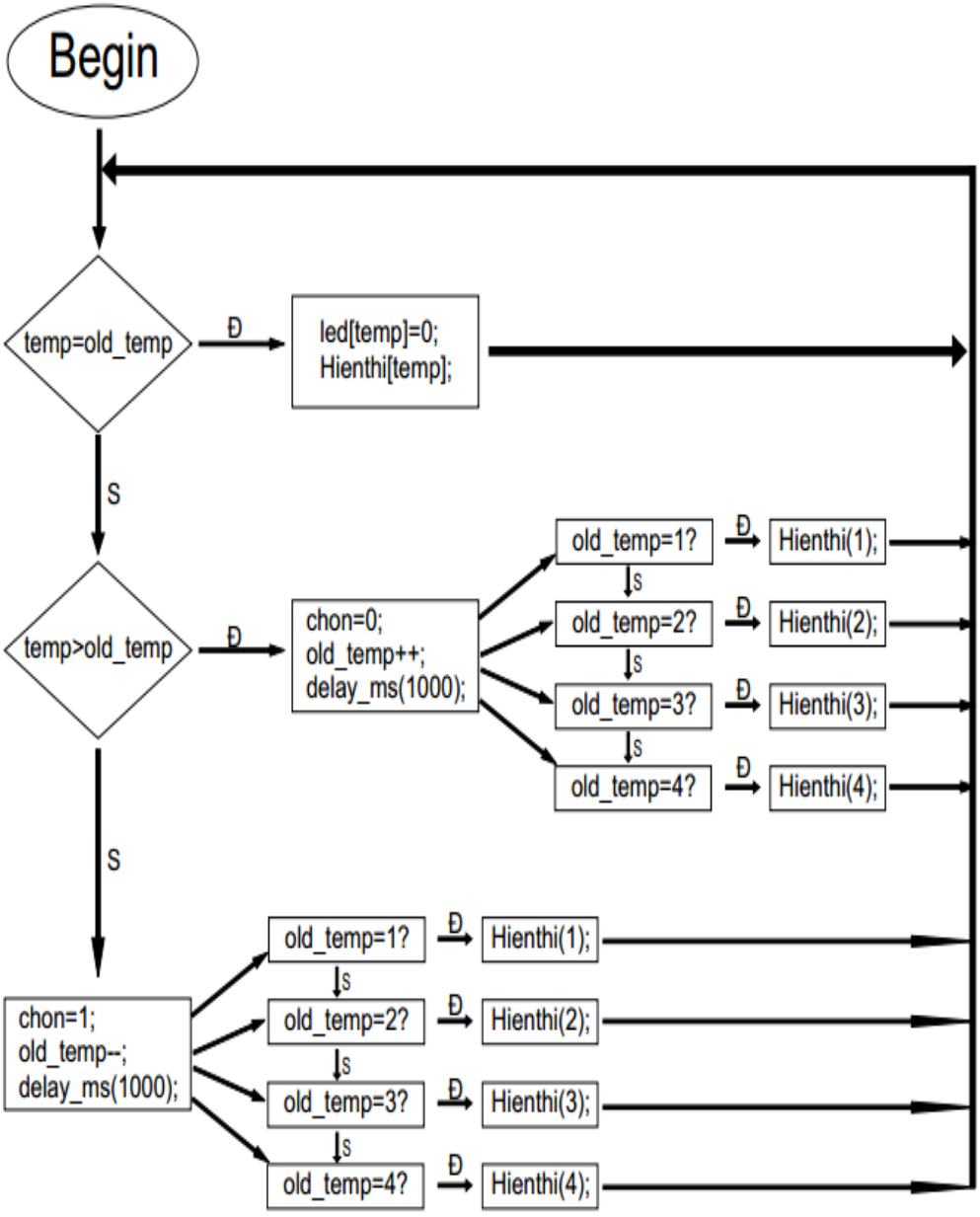
KHỐI PHÍM BẤM

3.4. Khối nguồn



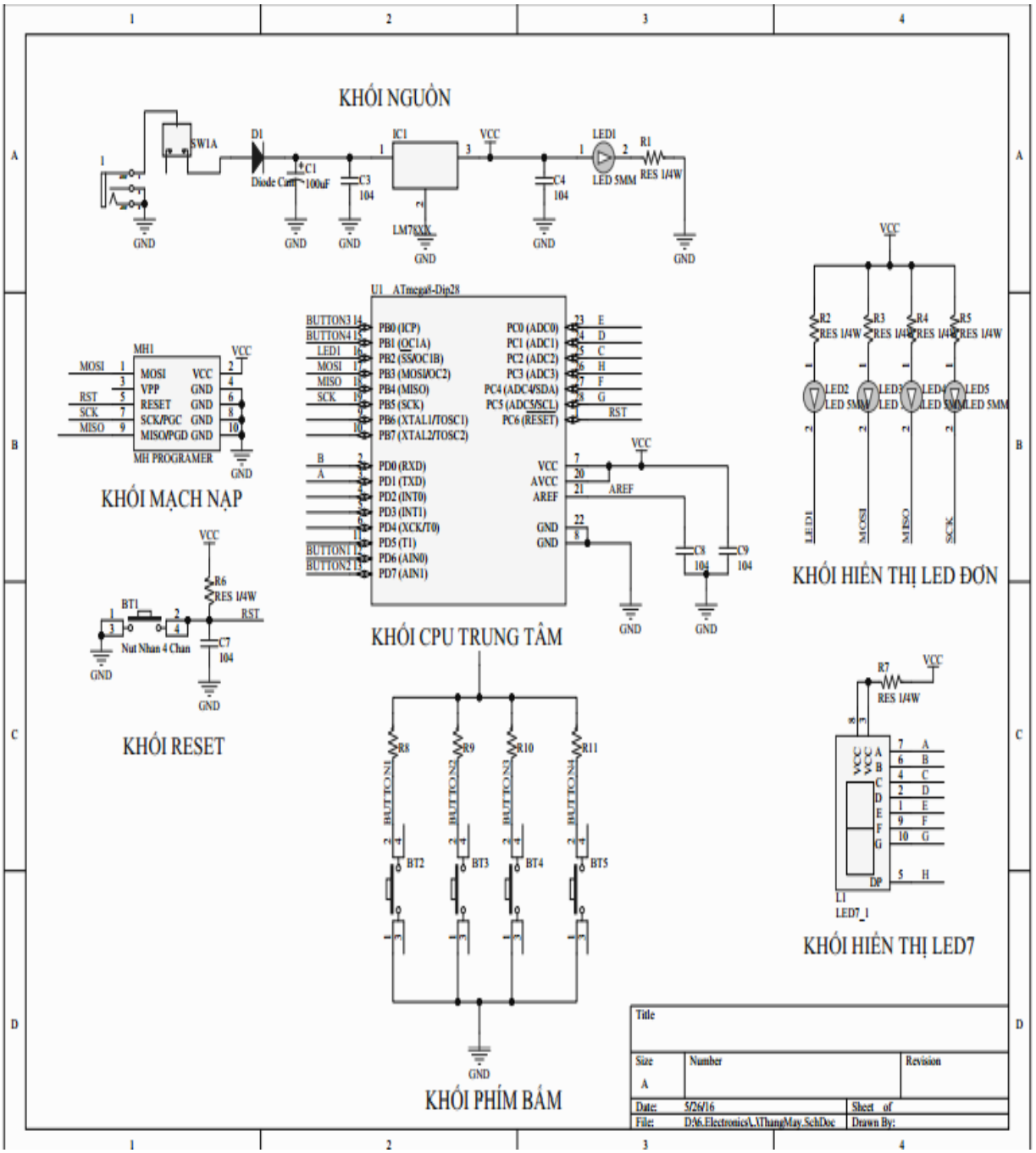
Chương 4: Xây dựng thuật toán và thiết kế mạch

4.1. Lưu đồ thuật toán:

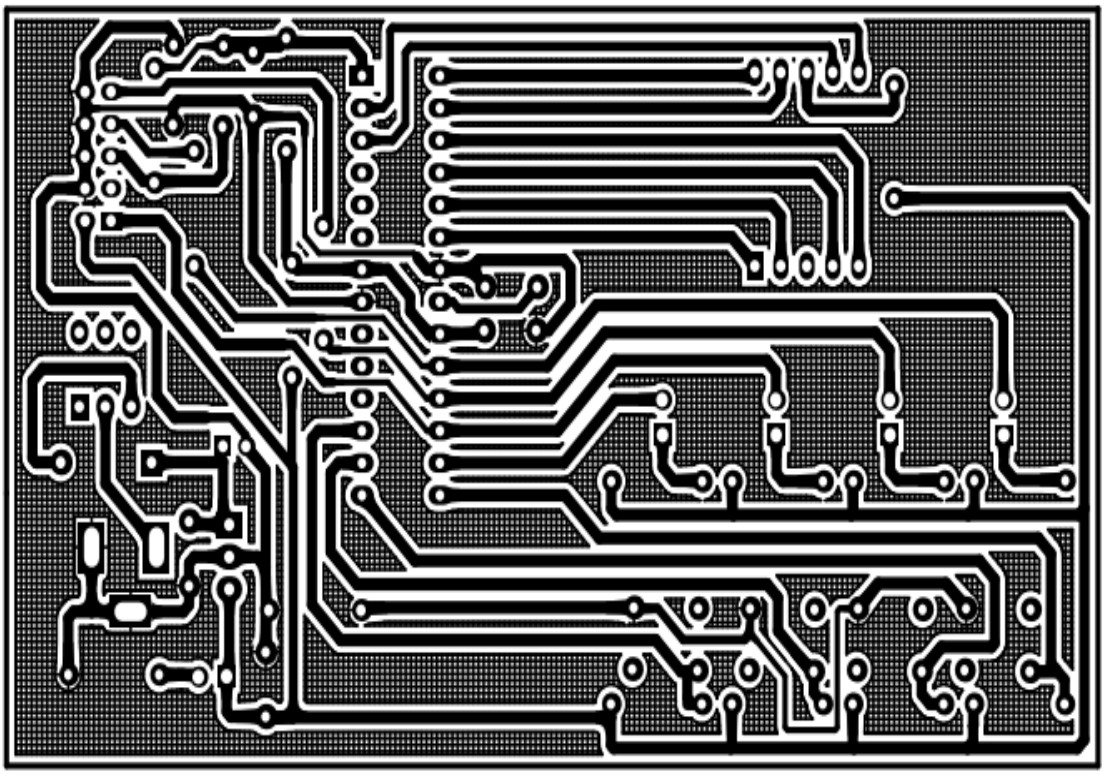


4.2. Thiết kế mạch:

a) Mạch nguyên lí:


KHỐI NGUỒN
SW1A
D1
Diode Cam
C1
100uF
C3
104
IC1
LM78XX
VCC
C4
104
LED1
LED 5MM
R1
RES 1/4W
GND
KHỐI MẠCH NẠP
MH1
MOSI
VPP
RESET
SCK/PGC
MISO/PGD
MH PROGRAMER
KHỐI RESET
BT1
Nut Nhan 4 Chan
R6
RES 1/4W
RST
C7
104
U1 ATmega8-Dip28
PB0 (ICP)
PB1 (OC1A)
PB2 (SS/OC1B)
PB3 (MOSI/OC2)
PB4 (MISO)
PB5 (SCK)
PB6 (XTAL1/TOSC1)
PB7 (XTAL2/TOSC2)
PD0 (RXD)
PD1 (TXD)
PD2 (INT0)
PD3 (INT1)
PD4 (XCK/T0)
PD5 (T1)
PD6 (AIN0)
PD7 (AIN1)
PC0 (ADC0)
PC1 (ADC1)
PC2 (ADC2)
PC3 (ADC3)
PC4 (ADC4/SDA)
PC5 (ADC5/SCL)
PC6 (RESET)
AVCC
AREF
C8
104
C9
104
KHỐI CPU TRUNG TÂM
R2
R3
R4
R5
RES 1/4W
LED2
LED3
LED4
LED5
LED 5MM
KHỐI HIỂN THỊ LED ĐƠN
R7
RES 1/4W
LED7_1
KHỐI HIỂN THỊ LED7
R8
R9
R10
R11
BT2
BT3
BT4
BT5
KHỐI PHÍM BẤM
Title
Size
A
Number
Revision
Date: 5/26/16
Sheet of
File: D:\6.Electronics\...\ThangMay.SchDoc
Drawn By:

b) Mạch in:



c) Liệt kê các linh kiên:

STT	Tên linh kiện	Giá trị	Số lượng
1	Atmega8		1
2	Nút bấm		5
3	Led 7 thanh		1
4	Led đơn		5
5	Tụ hóa	100MF	1
6	Tụ gốm	104	5
7	bjt	LM7805	1
8	Trở led	330	4
9	Trở nút bấm	10k	4
10	Trở reset	10k	1
11	Trở led báo nguồn	330	1

Mạch khi hoàn thiện:

4.3. Code chạy mô phỏng

```
Chip type               : ATmega8
Program type            : Application
AVR Core Clock frequency: 8.000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 256
*****************************************************/

#include <main.h>
//#include "var.h"

// Declare your global variables here
uint8_t
maled7[]={0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0x80,0x90,0x8
8,0x83,0xC6,0xA1,0x86,0x8E};

uint8_t temp,old_temp,cong,tru;

void LED7SEG(uint8_t data);
void Check_Key();
void Display_Led(uint8_t Local,uint8_t chon);
void Prog_Run();

void main(void)
{

// Khai Bao PORT B
PORTB=0xff;
DDRB=0xFC;
// Khai Bao PORT C
PORTC=0xff;
```

```
DDRC=0xFF;
// Khai Bao PORT D
PORTD=0xff;
DDRD=0x3F;
// Khai Bao TIMER 0
TCCR0=0x00;
TCNT0=0x00;
// Khai Bao TIMER 1
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
// Khai Bao TIMER 2
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
// Khai Bao Ngat Ngoai
MCUCR=0x00;
MCUCSR=0x00;
// Khai Bao Ngat Timer
TIMSK=0x00;
// Khai Bao UART
UCSRB=0x00;
// Khai Bao bo so sanh Analog
ACSR=0x80;
SFIOR=0x00;
// ADC initialization
```

```
// ADC Clock frequency: 1000.000 kHz
// ADC Voltage Reference: AVCC pin
//ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x00;

// Khai Bao SPI
SPCR=0x00;
// Khai Bao TWI
TWCR=0x00;

LED1=0;
LED7SEG(maled7[1]);
old_temp=temp=1;
while (1)
     {
       Prog_Run();
     }
}

void LED7SEG(uint8_t data)
{
   PORTD_1 = data&0x01;
   PORTD_0 = data&0x02;
   PORTC_2 = data&0x04;
   PORTC_1 = data&0x08;
   PORTC_0 = data&0x10;
   PORTC_4 = data&0x20;
   PORTC_5 = data&0x40;
   PORTC_3 = data&0x80;
}
void Check_Key()
{
 if(BT1==0)       {delay_ms(200); if(BT1==0)temp=1;}
 else if(BT2==0)  {delay_ms(200); if(BT2==0)temp=2;}
```

```
 else if(BT3==0)  {delay_ms(200); if(BT3==0)temp=3;}
 else if(BT4==0)  {delay_ms(200); if(BT4==0)temp=4;}
}
void Display_Led(uint8_t Local,uint8_t chon)
{
 if(chon==0)
 {
     switch(Local)
    {
      case 1 : {
                 LED1=0; LED7SEG(maled7[1]);
               }break;
      case 2 : {
                 LED1=1; LED2=0; LED7SEG(maled7[2]);
               }break;
      case 3 : {
                 LED2=1 ;LED3=0; LED7SEG(maled7[3]);
               }break;
      case 4 : {
                 LED3=1; LED4=0; LED7SEG(maled7[4]);
               }break;
    }
 }
 if(chon==1)
 {
      switch(Local)
    {
      case 1 : {
                 LED1=0; LED2=1; LED7SEG(maled7[1]);
               }break;
      case 2 : {
                 LED2=0; LED3=1; LED7SEG(maled7[2]);
               }break;
      case 3 : {
```

```
                LED3=0; LED4=1; LED7SEG(maled7[3]);
              }break;
        case 4 : {
                LED7SEG(maled7[4]);
              }break;
    }
 }
}
void Prog_Run()
{
 Check_Key();
 if(old_temp!=temp)
 {
    if(old_temp<temp)
    {
      Display_Led(old_temp,0);
      old_temp++;
      delay_ms(1000);
      cong=1;
      tru=0;
    }
    else
    {
      Display_Led(old_temp,1);
      old_temp--;
      delay_ms(1000);
      tru=1;
      cong=0;
    }
 }
 else
 {
  old_temp=temp;
  if(cong) Display_Led(old_temp,0);
```

```
 if(tru) Display_Led(old_temp,1);
 }
}
```

Chương V : Tổng kết

5.1 .Mức độ hoàn thành công việc và hướng phát triển.

1. ***Mức độ hoàn thành công việc.***

   Nhìn chung đã hoàn thành về mạch mô phỏng và chạy ổn định các khâu của bài tập lớn.

2. ***Hướng phát triển.***

   Có thể thêm vào phần cảm biến nguy hiểm và hệ thống cảnh báo và liên lạc ở ngoài..

5.2 .Các phần mềm đã sử dụng trong quá trình làm bài tập.

- Proteus version 8.0
- Codevison AVR 2.5.3

5.2. Tài liệu tham khảo.

Website:

http://hocavr.com

http://hoiquandientu.com

http://dientuvietnam.com

http://dientu.org

http://machdientu.net

Ebook:Datasheet : Atmeaga8.

Hết