ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỆN TỬ
----------o0o----------

BÁO CÁO BÀI TẬP LỚN VI XỬ LÍ MỞ RỘNG

MẠCH ĐO TẦN SỐ DÙNG MCU 8051

GVHD: TS. NGUYỄN LÝ THIÊN TRƯỜNG
SVTH: TRẦN MINH NHẬT
MSSV: 2014008

TP. HỒ CHÍ MINH, THÁNG 06 NĂM 2022

LỜI CẢM ƠN

Để hoàn thành bài báo cáo này, em xin bày tỏ lòng biết ơn sâu sắc tới tiến sĩ Nguyễn Lý Thiên Trường đã tạo điều kiện cũng như hỗ trợ để em hoàn thành báo cáo này.

Em cũng chân thành tỏ lòng biết ơn tới toàn thể thầy cô trong Khoa Điện – Điện tử, Đại học Bách Khoa TP.HCM, Đại học Quốc Gia TP.HCM đã tận tình truyền đạt kiến thức và nhiều kinh nghiệm.

Cuối cùng, xin chúc quý thầy, cô dồi dào sức khỏe và thành công trong sự nghiệp cao quý.

Tp. Hồ Chí Minh, ngày 09 tháng 06 năm 2022.

Sinh viên

Trần Minh Nhật

TÓM TẮT NỘI DUNG

Bài báo cáo này trình bày về mạch ứng dụng dùng vi điều khiển 8051 đo tần số và duty cycle của tín hiệu xung vuông. Với tần số đo được có giá trị trong khoảng 1Hz đến 100Hz được cấp vào chân P3.2 (\INT0) của vi điều khiển 8051.

Kết quả đo được sẽ hiển thị ra màn hình LCD 16 x 2 với hàng 1 là giá trị tần số đo được, và hàng 2 là giá trị của chu kỳ nhiệm vụ. Dùng Fosc = 12MHz.

MỤC LỤC

DANH SÁCH HÌNH MINH HỌA

1. GIỚI THIỆU ĐỀ TÀI

1.1 Tổng quan

Sự phát triển chung của khoa học kĩ thuật trên toàn thế giới ngày càng trở nên quan trọng bằng nền sản xuất đa dạng, số lượng rất lớn. Nền sản xuất này không chỉ đòi hỏi số lượng lớn người lao động và còn rất chú trọng vấn đề tự động hóa trong quá trình sản xuất, nghiên cứu để đẩy nhanh quá trình sản xuất, phục vụ nhu cầu của con người. Kèm theo sự phát triển đó là sự xuất hiện rộng rãi của các chip vi xử lí, IC lập trình đã đẩy vấn đề tự động hóa nên một bước tiến cao hơn. Các ứng dụng, các nghiên cứu về nó cũng được chú trọng phát triển theo.

Tần số là chủ đề quen thuộc và quan trọng trong lĩnh vực điện – điện tử. Trong lĩnh vực này không thể bỏ qua và lướt bỏ thông số tần số được, vì nó có vai trò quan trọng quyết định đến sự phát triển của lĩnh vực điện tử. Hiện nay các máy đếm tần số cao với độ chính xác cực cao đã rất phát triển, đặc biệt các bộ đếm này có độ ổn định rất tốt. Tuy nhiên do sự hạn chế về băng thông của các mạch chúng ta cần khảo sát nên dải tần số đo có thể bị hạn chế, cũng như các vấn đề về kinh tế mà chúng ta cần một thiết bị, một bộ đo đơn giản để khảo sát nhanh tần số để có thể đánh giá nhanh hệ thống hoặc khảo sát nhanh các đặc tính.

Trong khuôn khổ của môn học vi xử lí và hiểu biết của mình, em đã tìm hiểu và thực hiện đề tài “Mạch đo tần số” sử dụng vi điều khiển AT89C51 để đo tần số điện và chu kì nhiệm vụ của một xung vuông.

1.2 Nhiệm vụ đề tài

Mục tiêu của đề tài là thiết kế phần cứng, lập trình các giải thuật để hiện thực mô hình mạch tính toán đo đạc tần số và chu kỳ nhiệm vụ

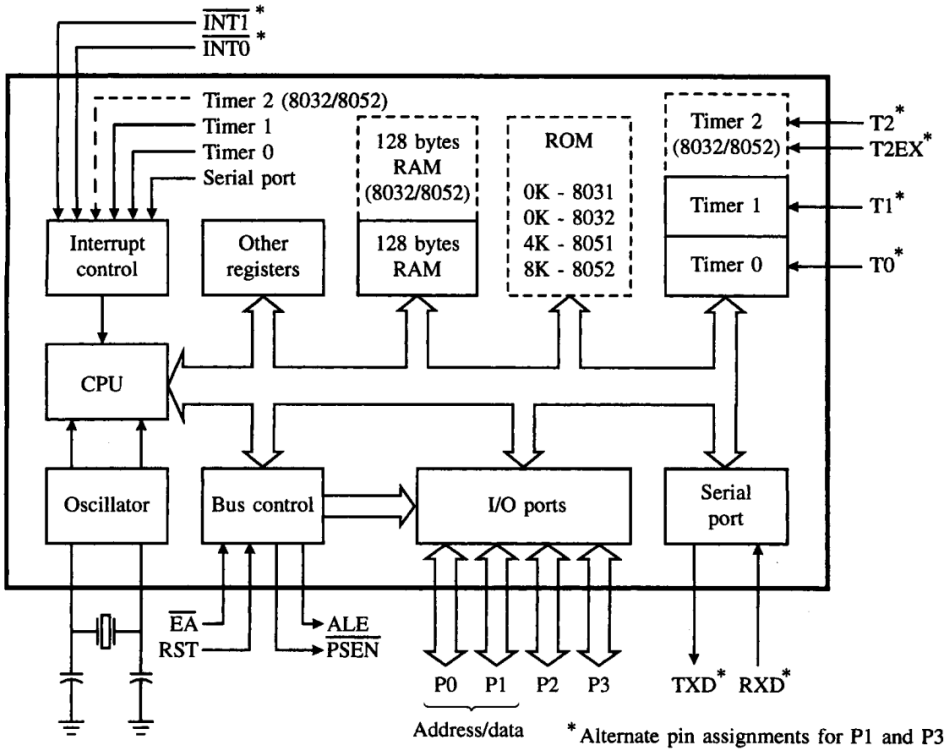
Một số vấn đề đặt ra:

- Tìm hiểu nguyên lý hoạt động và lí thuyết của vi điều khiển AT89C51.
- Tìm hiểu về LCD 16 x 2 và các tập lệnh điều khiển cũng như cách giao tiếp LCD với vi điều khiển.
- Tìm hiểu về tập lệnh hợp ngữ của dòng vi điều khiển MCS51.
- Tìm hiểu và tham khảo về các thuật toán liên quan đến chủ đề.

2. LÝ THUYẾT LIÊN QUAN ĐẾN ĐỀ TÀI

2.1 Giới thiệu tổng quan về họ vi điều khiển Intel MCS-51

Intel MCS-51 là vi điều khiển đơn tinh thể kiến trúc Hardvard, lần đầu tiên được sản xuất bởi Intel năm 1980, để dùng trong các hệ thống nhúng. Họ vi điều khiển này là vi điều khiển 8 bit, chế tạo theo công nghệ CMOS chất lượng cao và yêu cầu công suất thấp.



Hình 2.1 Cấu tạo của 8051

Kiến trúc cơ bản bên trong 8051 bao gồm các đặt tính sau:

- Một ALU 8-bit, một thanh tích lũy và một thanh ghi 8-bit, do đó nó là một vi điều khiển 8-bit
- Bus dữ liệu 8-bit: có thể truy cập 8 bits dữ liệu để hoạt động
- Bus địa chỉ 16-bit: có thể truy cập 2^{16} địa chỉ nhớ 64KB (65536 địa chỉ) cho mỗi bộ nhớ RAM/ROM
- RAM onchip: 128 bytes bộ nhớ dữ liệu
- ROM onchip: 4KB bộ nhớ chương trình
- 32 chân I/O riêng biệt (4 port mỗi nhóm 8 chân I/O) có thể được truy cập riêng lẽ
- Hai bộ định thời/đếm 16-bit
- Đơn vị thu/phát bất đồng bộ phổ biến UART song công

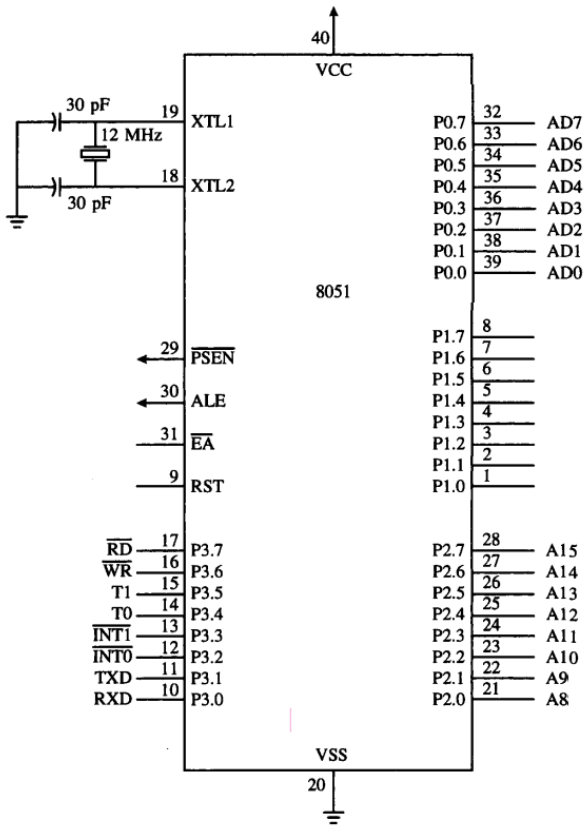
- 6 nguồn ngắt với 2 mức ưu tiên
- Chế độ tiết kiệm năng lượng

Lập trình trên 8051:

- Có thể sử dụng các trình biên dịch từ ngôn ngữ lập trình C
- Phần lớn sử dụng ngôn ngữ C và Assembly để lập trình cho 8051

2.2 Sơ lược về các chân kết nối của IC 8051

Vi điều khiển 8051 có tất cả 40 chân có chức năng như các đường xuất nhập. Trong đó 24 chân có tác dụng kép, mỗi chân có thể hoạt động như các đường xuất nhập hoặc như các tín hiệu điều khiển hoặc là thành phần của bus dữ liệu.



Hình 2.2 Sơ đồ chân của 8051

Chức năng của các chân của AT89C51

- **Port 0:** là port có chức năng kép, là các chân số 32-39 của IC 8051. Trong thiết kế nhỏ không dùng bộ nhớ mở rộng thì **Port 0** có chức năng xuất nhập

I/O thông thường. Đối với thiết kế lớn hơn có kết nối bộ nhớ bên ngoài thì **Port 0** có chức năng như byte thấp của bus địa chỉ và bus dữ liệu

- **Port 1:** là port có chức năng I/O xuất nhập thông thường giao tiếp với các ngoại vi bên ngoài, là các chân số 1-8 của IC8051
- **Port 2:** là port có chức năng kép (chân 21-28) được dùng như các đường xuất nhập I/O hoặc đóng vai trò là byte cao của bus địa chỉ đối với thiết kế lớn có dùng các bộ nhớ mở rộng
- **Port 3:** cũng là port có chức năng kép (chân 10-17) được dùng như các đường xuất nhập I/O thông thường và mỗi chân của **Port 3** có chức năng đặc biệt riêng

Bit	Name	Bit Address	Alternate Function
P3.0	RXD	B0H	Receive data for serial port
P3.1	TXD	B1H	Transmit data for serial port
P3.2	$\overline{\text{INT0}}$	B2H	External interrupt 0
P3.3	$\overline{\text{INT1}}$	B3H	External interrupt 1
P3.4	T0	B4H	Timer/counter 0 external input
P3.5	T1	B5H	Timer/counter 1 external input
P3.6	$\overline{\text{WR}}$	B6H	External data memory write strobe
P3.7	$\overline{\text{RD}}$	B7H	External data memory read strobe
P1.0	T2	90H	Timer/counter 2 external input
P1.1	T2EX	91H	Timer/counter 2 capture/reload

Hình 2.3 Chức năng đặc biệt của các chân Port 3

Ngõ tín hiệu $\overline{PSEN}$ (Program Store Enable)

IC8051 có 4 bus điều khiển tín hiệu. Chân $\overline{PSEN}$ là chân số 29 – tín hiệu ngõ ra có tác dụng cho phép đọc bộ nhớ chương trình mở rộng và thường được nối với chân $\overline{OE}$ (output enable) của EPROM cho phép đọc các byte mã lệnh

$\overline{PSEN}$ ở mức thấp trong thời gian lấy lệnh. Các mã nhị phân của chương trình đọc từ EPROM qua bus dữ liệu và được chốt vào thanh ghi bên trong 8051 để giải mã lệnh. Khi thi hành chương trình thì $\overline{PSEN}$ sẽ ở mức cao

Ngõ tín hiệu điều khiển ALE (Address Latch Enable)

ALE là tín hiệu ngõ ra ở chân 30 có chức năng phân kênh địa chỉ và dữ liệu. Khi Port 0 được sử dụng ở chế độ mở rộng bộ nhớ - byte thấp địa chỉ thì ALE là tín hiệu chốt địa chỉ ra bên ngoài trong nửa đầu chu kì bộ nhớ. Nửa sau chu kì bộ nhớ, Port 0 sẽ có chức năng là ngõ vào và ngõ ra dữ liệu giúp chuyển dữ liệu

ALE có tốc độ bằng 1/6 tần số dao động trên vi điều khiển nên có thể dùng để tạo tín hiệu xung clock cho các ứng dụng khác của hệ thống.

Ngõ tín hiệu $\overline{EA}$ cho phép truy xuất ngoài (Enable Access)

Đây là tín hiệu ngõ vào ở chân 31 thường được mắc với mức cao (+5V) hoặc mức thấp (đất). Nếu chân $\overline{EA}$ được nối với mức cao thì 8051 sẽ thực thi chương trình trong ROM nội trong khoảng 4K bộ nhớ thấp. Còn ở mức thấp (tích cực) thì 8051 sẽ thực thi chương trình từ bộ nhớ ngoài (bộ nhớ mở rộng)

Ngõ tín hiệu RST (Reset)

Tín hiệu RST là chân số 9 dùng để reset lại IC8051. Khi tín hiệu này được đưa lên mức cao ít nhất 2 chu kì máy thì 8051 sẽ đưa các thanh ghi về các giá trị mặc định để sẵn sàng khởi động lại hệ thống

Ngõ vào bộ dao động trên chip

XTAL1 và **XTAL2** là hai ngõ vào dao động trên chip 8051, kết hợp với thạch anh (12MHz). Khi sử dụng 8051 người thiết kế cần ghép nối thêm tụ, thạch anh.

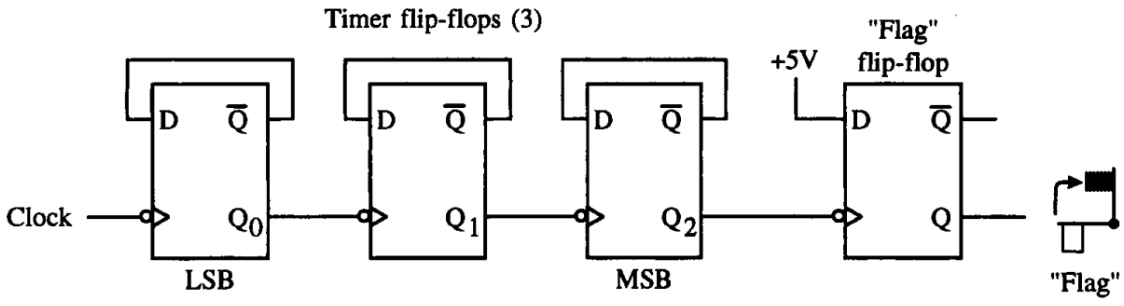
Nguồn cho IC8051

Nguồn cấp cho IC8051 tại chân 40 (V_{CC}) và chân 20 (V_{SS})

2.3 Hoạt động của bộ định thì (đếm)

Giới thiệu

Timer được xem như một chuỗi các flip-flop chia đôi tần số được mắc nối tiếp với nhau → **Bộ đếm lên**



Hình 2.4 Timer là chuỗi các flip-flop chia đôi tần số

Ngõ ra của tầng cuối làm xung nhịp cho flip-flop báo tràn của timer. Giá trị nhị phân trong các flip-flop của timer có thể xem như đếm số xung nhịp từ khởi động timer. Ví dụ timer 8 bit sẽ đếm từ 00H đến FFH, còn timer 16 bit sẽ đếm từ 0000H đến FFFFH. Cờ báo tràn (TF0, TF1) sẽ bật lên 1 tại thời điểm nội dung bộ đếm chuyển trạng thái về lại giá trị 0

IC8051 có 2 timer: Timer 0 và Timer 1. Mỗi timer là một bộ đếm **tối đa 16 bit**. Mỗi bộ định thời có 3 chức năng làm việc:

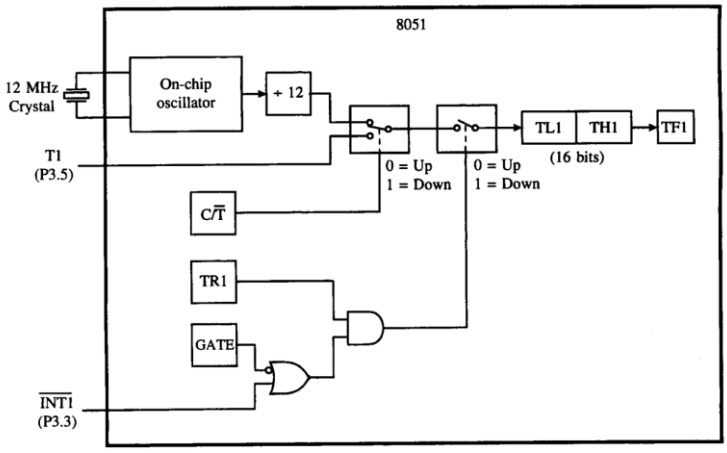
- **Timer:** được dùng để tạo khoảng thời gian trễ, đo bề rộng xung. Dùng xung clock dao động nội (dao động từ thạch anh)
- **Counter:** đếm sự kiện. "Sự kiện " chuyển tiếp từ 1 → 0 ở chân T0 (P3.4) hoặc T1 (P3.5). Dùng nguồn xung clock từ dao động bên ngoài thông qua chân T0 hoặc T1. Chẳng hạn như đếm số người vào phòng, đếm sản phẩm, đếm số xung, đo tần số, số vòng quay động cơ,…
- **Tạo tốc độ baud cho truyền dữ liệu nối tiếp SPI**

Thanh ghi TMOD (Timer Mode) – Địa chỉ byte 89H

GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0

Thanh ghi TMOD không có địa chỉ hóa từng bit và được cài đặt bằng phần mềm các chế độ hoạt động tùy theo nhu cầu của người thiết kế

Bit	Tên	Timer	Mô tả
7	GATE	1	Khi GATE=1, Timer chỉ chạy khi $\overline{INT1}$=1
6	C/$\overline{T}$	1	1 là đếm sự kiện / 0 là định thì
5	M1	1	Bit 1 chọn chế độ
4	M0	1	Bit 0 chọn chế độ
3	GATE	0	Khi GATE=1, Timer chỉ chạy khi $\overline{INT0}$=1
2	C/$\overline{T}$	0	1 là đếm sự kiện / 0 là định thì
1	M1	0	Bit 1 chọn chế độ
0	M0	0	Bit 0 chọn chế độ



Hình 2.5 Cấu trúc của Timer 1

Thanh ghi điều khiển Timer (TCON)

Thanh ghi TCON chứa các bit trạng thái và các bit điều khiển cho timer 0 và timer 1, 4 bit cao của TCON (TCON.4 – TCON.7) được sử dụng để bật và tắt timer (TR0, TR1) hoặc tín hiệu báo tràn timer (TF0, TF1)

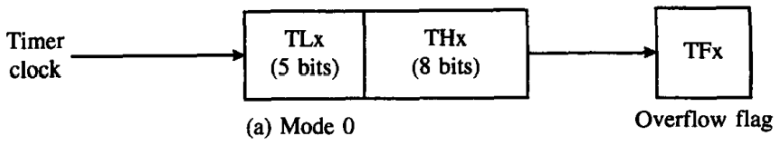
Bit	Kí hiệu	Địa chỉ	Mô tả
TCON.7	TF1	8FH	Cờ báo tràn timer 1
TCON.6	TR1	8EH	Bit điều khiển timer 1 chạy
TCON.5	TF0	8DH	Cờ báo tràn timer 0
TCON.4	TR0	8CH	Bit điều khiển timer 0 chạy
TCON.3	IE1	8BH	Cờ báo ngắt ngoài 1
TCON.2	IT1	8AH	Bắt cạnh xuống khi ngắt ngoài 1
TCON.1	IE0	89H	Cờ báo ngắt ngoài 0
TCON.0	IT0	88H	Bắt cạnh xuống khi ngắt ngoài 0

Các chế độ hoạt động của timer

Các bit thiết lập chế độ hoạt động

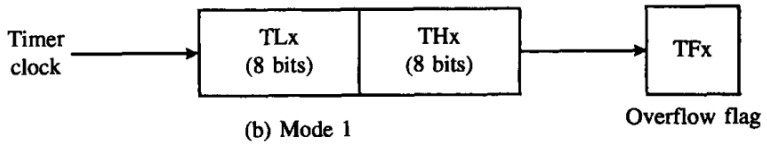
M1	M0	MODE	Chế độ hoạt động
0	0	0	Chế độ Timer/Counter 13-bit 8-bit THx + 5-bit (thấp) TLx (x=0 hoặc x=1)
0	1	1	Chế độ Timer/Counter 16-bit 8-bit THx + 8-bit TLx (x = 0 hoặc x = 1)
1	0	2	Chế độ Timer/Counter 8-bit tự nạp lại giá trị đầu THx lưu giá trị nạp vào TLx mỗi khi tràn
1	1	3	Chế độ tách Timer

- Mode 0: Chế độ Timer/Counter 13-bit



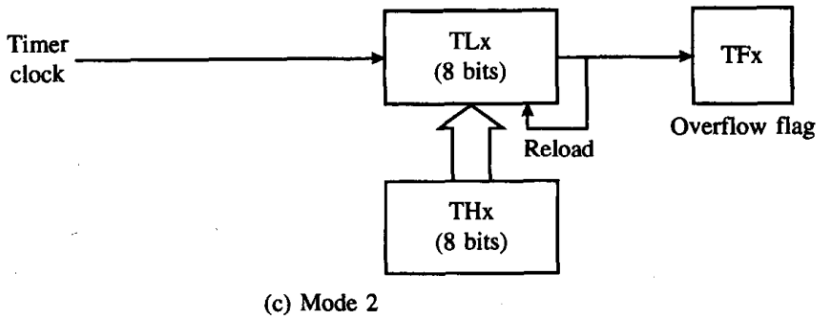
Hình 2.6 Timer mode 0

- Mode 1: Chế độ Timer/Counter 16-bit



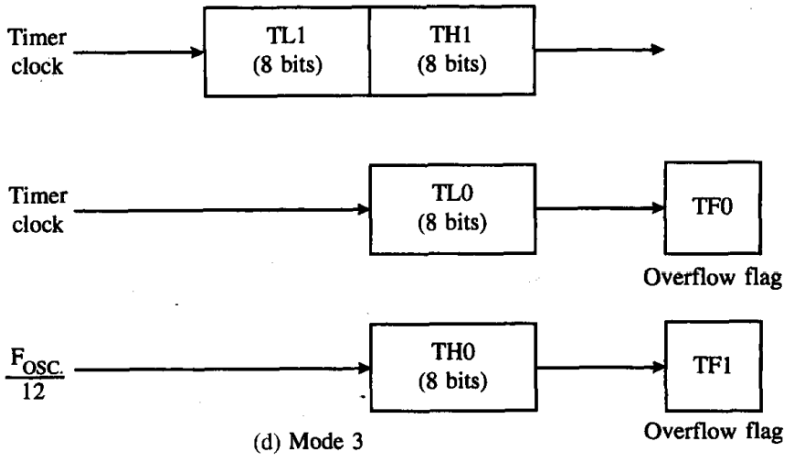
Hình 2.7 Timer mode 1

- Mode 2: Chế độ Timer/Counter 8-bit tự nạp lại giá trị ban đầu



Hình 2.8 Timer mode 2

- Mode 3: Chế độ tách Timer



Hình 2.9 Timer mode 3

2.4 Hoạt động của bộ ngắt (interrupt)

Giới thiệu

Khi có ngắt xảy ra, chương trình chính sẽ tạm dừng và chuyển sang chương trình phục vụ ngắt.

Ngắt đóng vai trò quan trọng trong việc thiết kế và cài đặt các ứng dụng vi điều khiển. Chúng cho phép hệ thống bất đồng bộ với một sự kiện và thực hiện sự kiện đó trong khi chương trình khác đang thực thi.

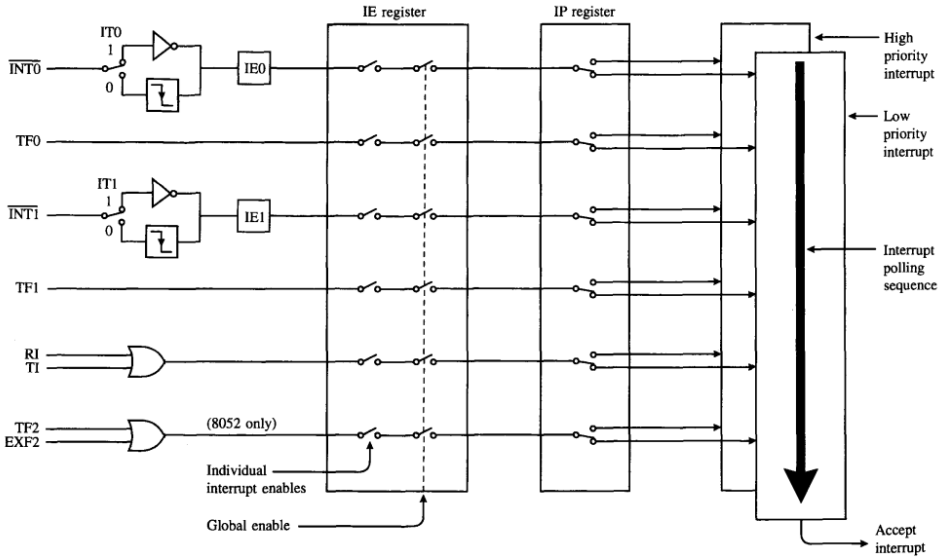
Một hệ thống được điều khiển bằng ngắt giống như làm nhiều việc đồng thời. Do CPU không thể thực thi cùng một lúc hơn một lệnh. Nhưng nó có thể tạm dừng việc thực thi một chương trình này để thực thi một chương trình khác, rồi quay lại thực thi chương trình trước đó. Theo cách này, ngắt giống như một chương trình con, nhưng có sự khác biệt là các điều kiện ngắt xảy ra không biết trước và có thể xảy ra bất kì lúc nào, chúng ta sẽ không biết ngắt xảy lúc nào và khi nào chương trình chính bị ngắt quãng.

ISR (Interrupt Service Routine) là chương trình phục vụ ngắt. Khi có ngắt xảy ra, chương trình chính tạm dừng và hệ thống thực hiện ISR. Sau khi ISR thực thi đáp ứng ngắt thì kết thúc bằng lệnh RETI để trở về lại chương trình chính và tiếp tục.

Tổ chức ngắt của IC8051

Các nguồn ngắt ở IC8051: 2 nguồn ngắt ngoài, 2 ngắt timer và một ngắt port nối tiếp. Tất cả các ngắt đều bị cấm sau khi reset hệ thống và được cho phép bằng phần mềm

- Ngắt ngoài 0 xảy ra khi: có cạnh xuống hoặc mức 0 trên chân $\overline{INT0}$
- Ngắt ngoài 1 xảy ra khi: có cạnh xuống hoặc mức 0 trên chân $\overline{INT1}$
- Ngắt timer 0 xảy ra khi: timer 0 bị tràn hay TF0=1
- Ngắt timer 1 xảy ra khi: timer 1 bị tràn hay TF1=1
- Ngắt cổng nối tiếp xảy ra khi: hoàn tất truyền dữ liệu (bộ đệm truyền rỗng) hoặc hoàn tất nhận dữ liệu (bộ đệm nhận đầy)



Hình 2.10 Cấu trúc ngắt của 8051

Cho phép và cấm ngắt

Mỗi nguồn ngắt được cho phép hoặc cấm từng ngắt thông qua thanh ghi chức năng đặc biệt tên là IE (Interrupt Enable) tại địa chỉ A8H. Đây là thanh ghi có địa chỉ hóa từng bit

EA	-	ET2	ES	ET1	EX1	ET0	EX0

EA: Cho phép ngắt toàn cục

ET2: Cho phép ngắt timer 2

ES: Cho phép ngắt cổng nối tiếp

ET1: Cho phép ngắt timer 1

EX1: Cho phép ngắt ngoài 1

ET0: Cho phép ngắt timer 0

EX0: Cho phép ngắt ngoài 0

Ưu tiên ngắt (Interrupt Priority)

-	-	PT2	PS	PT1	PX1	PT0	PX0

PT2 : Ưu tiên ngắt timer 2

PS: Ưu tiên ngắt cổng nối tiếp

PT1: Ưu tiên ngắt timer 1

PT0: Ưu tiên ngắt timer 0

PX0: Ưu tiên ngắt ngoài 0

Xử lí ngắt (Processing Interrupt)

Khi ngắt xảy ra và CPU chấp thuận:

- Cờ ngắt tương ứng được thiết lập
- Hoàn tất thực thi lệnh hiện tại
- Cất PC vào stack
- Tải vector ngắt vào PC – vector ngắt là nơi mà ISR thực thi lệnh
- Cờ báo ngắt tự động xóa, ngoại trừ RI và TI
- Thực thi chương trình phục vụ ngắt
- ISR kết thúc với lệnh RETI, trả giá trị PC cũ từ stack và tiếp tục thực thi chương trình chính

Các vector ngắt (Interrupt Vectors)

Ngắt thứ	Ngắt	Địa chỉ ISR	Độ ưu tiên
0	**Ngắt ngoài 0**	**0003H**	**Cao nhất**
1	**Ngắt timer 0**	**000BH**	
2	**Ngắt ngoài 1**	**0013H**	
3	**Ngắt timer 1**	**001BH**	
4	**Ngắt nối tiếp**	**0023H**	**Thấp nhất**

2.5 LCD và giao tiếp LCD với 8051

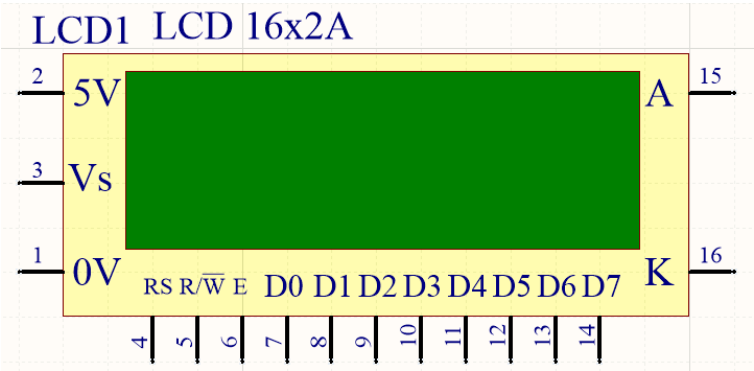
2.5.1 Cấu tạo của LCD

LCD (Liquid Crystals Display) – màn hình tinh thể lỏng , cơ sở vật lí để LCD có thể hiển thị được thông tin, hình ảnh chính là do đặc tính của vật liệu chế tạo nên LCd, tức là Liquid Crytals (thạch anh lỏng). Các tinh thể bình thường chúng ở thể rắn với sự định hướng đặc biệt. Tuy nhiên ở đây các tinh thể lỏng được cấu trúc từ các tinh thể động. Các tinh thể này có thể điều chỉnh bởi một điện trường đây là cách để điều khiển chất lỏng thay đổi từ trong suốt đến trạng thái mờ đục

LCD gồm 2 bề mặt dạng rãnh, giữa 2 bề mặt này là 1 lớp Thạch Anh lỏng (Liquid Crystal):

- Để có 1 điểm tối trên LCD: ánh sáng phát ra từ bên trong LCD sẽ đi qua bề mặt rãnh thứ nhất (lớp lọc đơn cực), sau đó ánh sáng đi qua lớp Liquid Cristal (lớp này được phân cực nên ánh sáng qua nó mà không bị xoắn), sau đó ánh sáng qua bề mặt rãnh thứ 2 lớp phân cực thứ 2 (lớp lọc đơn cực), ánh sáng không ló ra được khỏi lớp này (bị chặn lại hoàn toàn) → ta thấy 1 điểm tối trên màn hình LCD.
-
- Để có 1 điểm sáng trên LCD: quá trình đi tương tự nhưng khác ở chỗ ánh sáng qua lớp Liquid Cristal không được phân cực nên ánh sáng bị xoắn 90 độ, nhờ thế mà đi qua được bề mặt rãnh thứ 2 (lớp lọc đơn cực) →ta thấy 1 điểm sáng trên LCD.

2.5.2 Chức năng các chân của LCD



Hình 2.11 Các chân kết nối của LCD

Chân	Kí hiệu	Chức năng	Mô tả
1	VSS	Nguồn	GND
2	VDD	Nguồn	Nguồn (5V)
3	VEE	Điều khiển	Điều chỉnh độ tương phản
4	RS	Điều khiển	RS=0: truy xuất lệnh; RS=1 truy xuất data
5	RW	Điều khiển	RW=1: truy xuất đọc; RW=0: truy xuất ghi
6	E	Điều khiển	E=1: cho phép truy xuất LCD E=0: cấm truy xuất LCD
7-14	D0-D7	Data/command	RS=1: data; RS=0: lệnh
15	LEDA	Anode LED	Anode LED nền
16	LEDK	Cathode LED	Cathode LED nền

- Ghi lệnh: RS=0, RW=0, E=1, D7-D0=mã lệnh
- Đọc lệnh: RS=0, RW=1, E=1, D7=BF:BF=1 LCD bận, BF=0 LCD rỗi
- Ghi data: RS=1, RW=0, E=1, D7-D0=data
- Đọc data: RS=1, RW=1, E=1, D7-D0=data

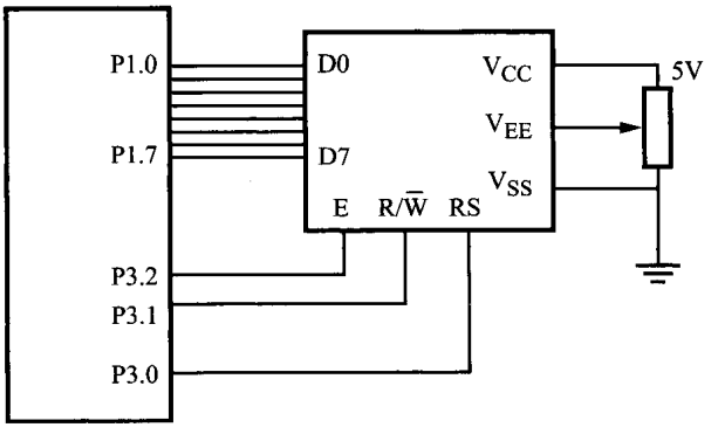
2.5.3 Tập lệnh của LCD

Các lệnh của LCD có thể chia thành 4 nhóm như sau:

- Các lệnh về hiển thị. Ví dụ: kiểu hiển thị (1 hàng/2 hàng), chiều dài dữ liệu..
- Chỉ định địa chỉ RAM nội.
- Nhóm lệnh truyền dữ liệu trong RAM nội.
- Các lệnh còn lại.

Mã lệnh (HEX)	Mô tả
0x01	Xóa hiển thị màn hình
0x02	Quay về vị trí home
0x04	Giảm con trỏ (dịch con trỏ sang trái)
0x06	Tăng con trỏ (dịch con trỏ sang phải)
0x05	Dịch màn hình sang phải
0x07	Dịch màn hình sang trái
0x08	Tắt hiển thị, tắt con trỏ
0x0A	Tắt hiển thị, bật con trỏ
0x0C	Bật hiển thị, tắt con trỏ
0x0E	Bật hiển thị, bật con trỏ
0x0F	Bật hiển thị, con trỏ chớp tắt
0x10	Dịch vị trí con trỏ sang trái
0x14	Dịch vị trí con trỏ sang phải
0x18	Dịch toàn bộ màn hình sang trái
0x1C	Dịch toàn bộ màn hình sang phải
0x80	Đưa con trỏ đến đầu dòng thứ 1
0xC0	Đưa con trỏ đến đầu dòng thứ 2
0x28	Chế độ 2 dòng, font 5x7 (D4-D7, 4 bit)
0x38	Chế độ 2 dòng

2.5.4 Giao tiếp LCD với 8051



Hình 2.12 LCD giao tiếp với 8051

Trường hợp ta không dùng lệnh đọc cờ BF để chờ LCD thực hiện xong lệnh, ta có thể gọi chương trình con DELAY để tạo thời gian trễ tối thiểu 50 μs, tốt nhất là 100 μs sau khi xuất lệnh ra LCD và phải tạo trễ 1520 μs khi xuất lệnh chuyển con trỏ về đầu dòng hoặc xóa LCD

Để khởi động màn hình LCD,ta thường thực hiện chuỗi các lệnh sau:

1. **Function set**: đặt cấu hình làm việc LCD kết nối 8/4 bit (DL), số dòng (N), ma trận 5x8 hay 5x10 dots (F)
2. **Clear display**: xóa toàn bộ màn hình (ghi lệnh mã 01H)
3. **Display control on**: bật màn hình on, con trỏ on
4. **Entry mode set**: chọn mode con trỏ dịch trái/phải,màn hình dịch trái/phải khi truy xuất data

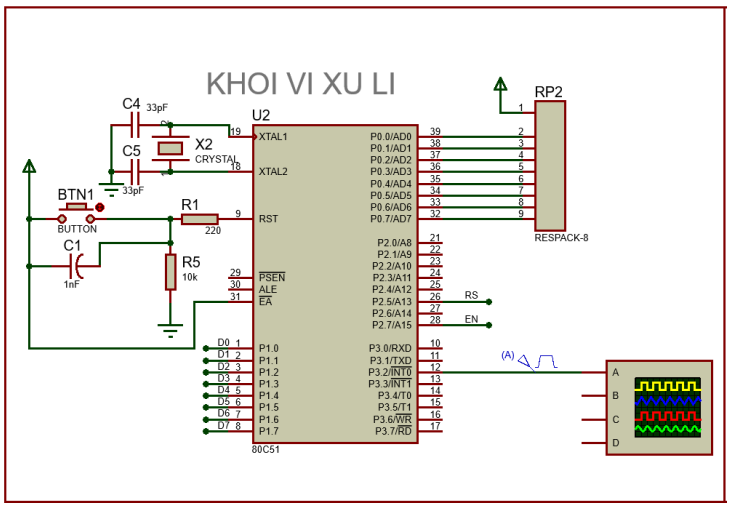
3. THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG

3.1 Các khối chức năng của hệ thống

3.1.1 Khối xử lí trung tâm

Khối xử lí trung tâm đóng vai trò đầu não của hệ thống, tiếp nhận các tín hiệu điều khiển và nhận xung vuông cần đo, thực hiện xử lí các phép toán, các mã lệnh nhận được và hiển thị kết quả lên *khối hiển thị kết quả* (màn hình LCD)

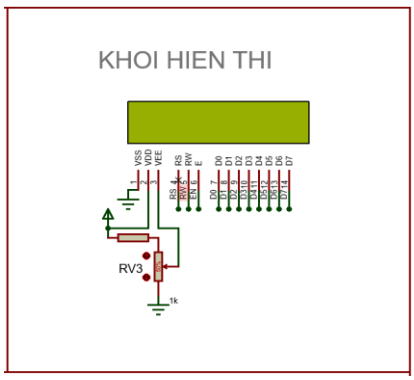
Khối xử lí trung tâm sử dụng vi điều khiển **AT89C51**



Hình 3.1 Khối vi xử lí

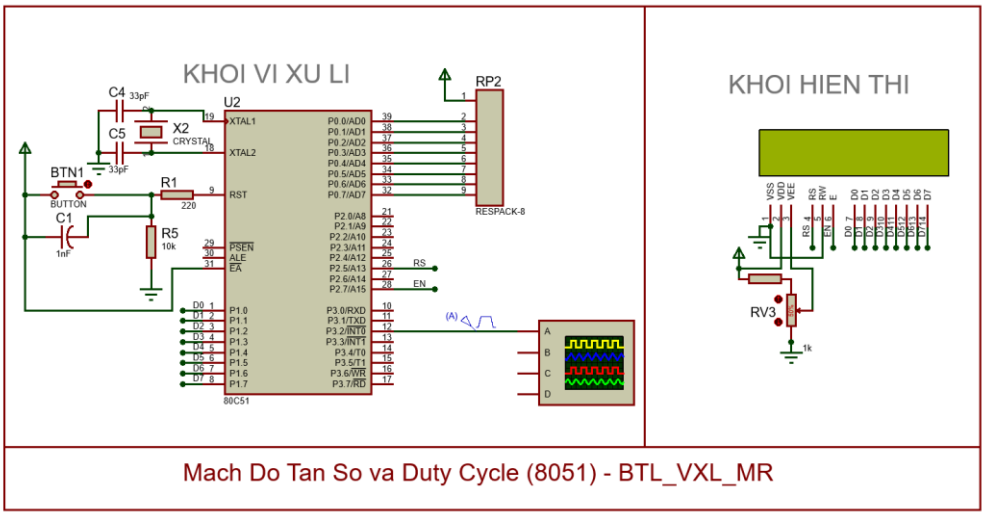
3.1.2 Khối hiển thị

Chứa màn hình hiển thị LCD 16×1 để hiển thị các các kết quả đo và tính toán. Yêu cầu đặt ra đối với khối hiển thị là phải thân thiện với người dùng, dễ dàng sử dụng và linh hoạt thay đổi nội dung



Hình 3.2 Khối hiển thị

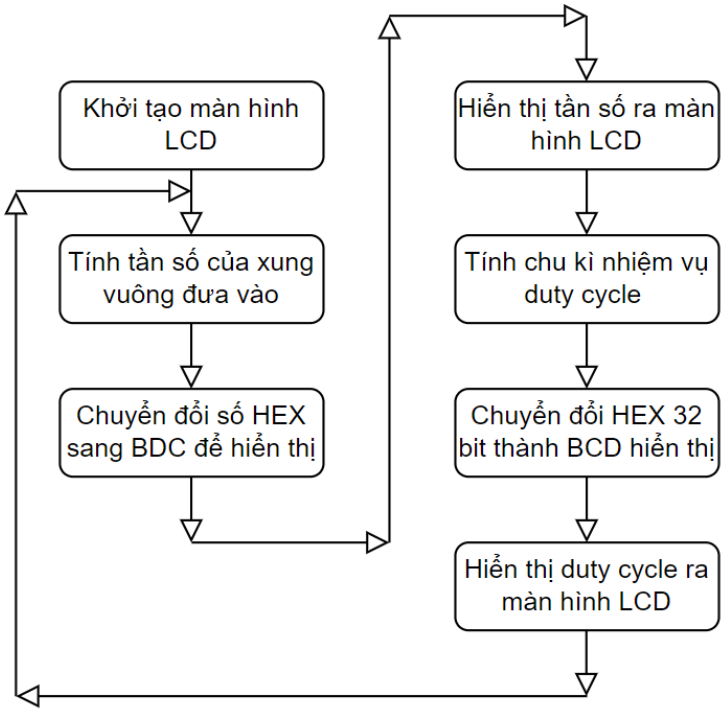
Sơ đồ mạch hoàn chỉnh



Hình 3.3 Sơ đồ mạch hoàn chỉnh

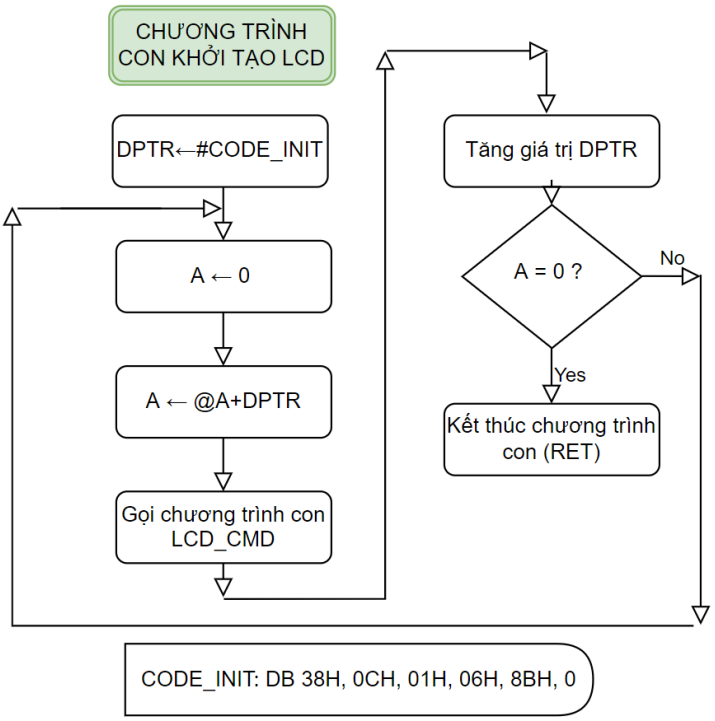
4. THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM

4.1 Lưu đồ giải thuật của chương trình chính – MAIN



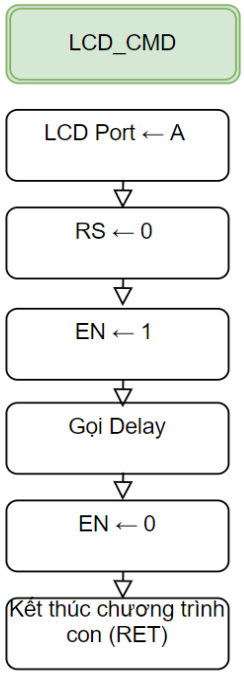
Hình 4.1 Lưu đồ giải thuật chương trình chính

4.2 Lưu đồ giải thuật của chương trình con khởi tạo LCD – LCD_INIT



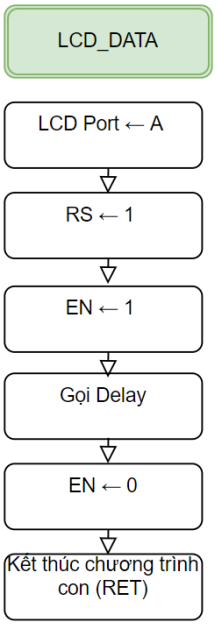
Hình 4.2 Lưu đồ giải thuật chương trình con khởi tạo LCD

4.3 Lưu đồ giải thuật của chương trình con gửi lệnh ra LCD – LCD_CMD



Hình 4.3 Lưu đồ giải thuật chương trình con gửi lệnh ra LCD

4.4 Lưu đồ giải thuật chương trình con gửi dữ liệu ra LCD – LCD_DATA

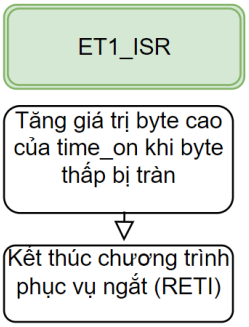


Hình 4.4 Lưu đồ giải thuật chương trình con gửi dữ liệu ra LCD

4.5 Lưu đồ giải thuật của chương trình phục vụ ngắt ngoài 0 – EX0_ISR

Hình 4.5 Lưu đồ giải thuật chương trình phục vụ ngắt ngoài 0

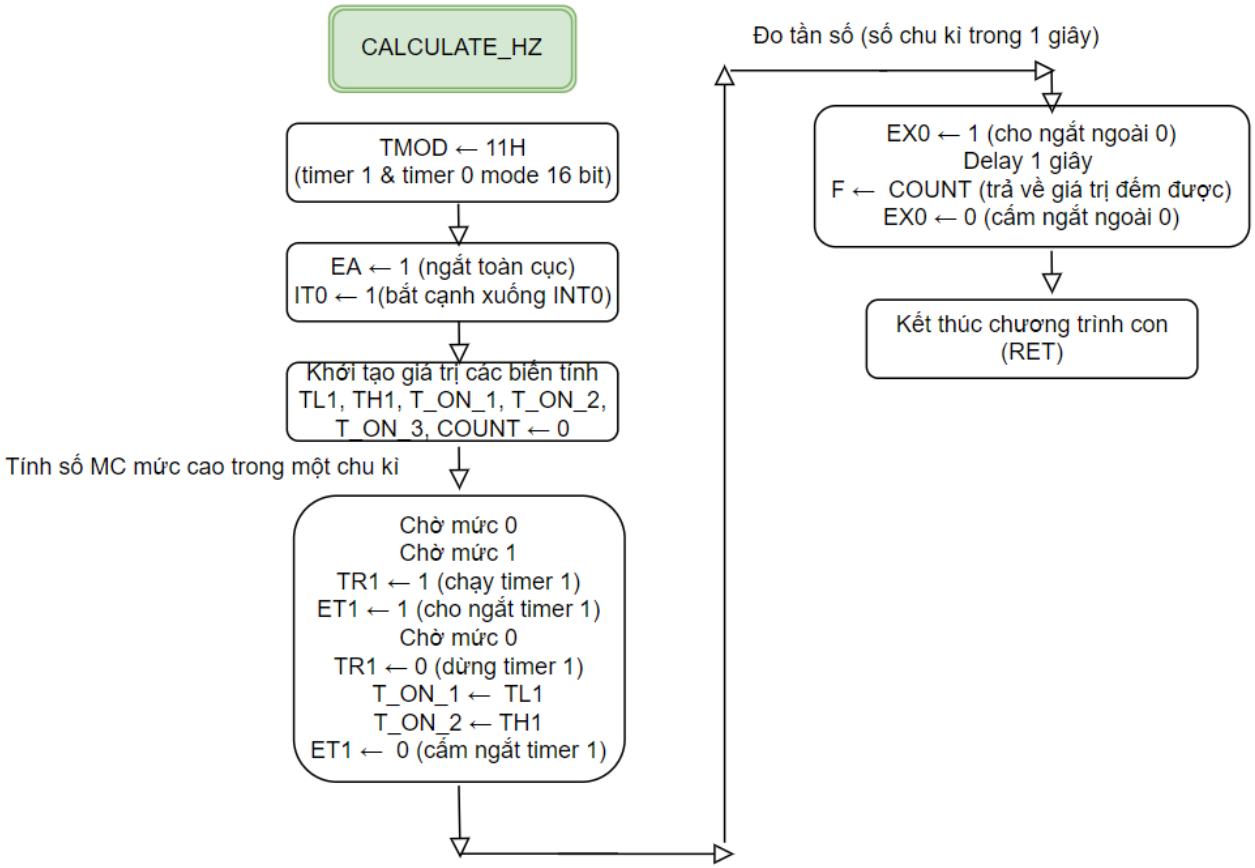
4.6 Lưu đồ giải thuật chương trình phục vụ ngắt timer 1 – ET1_ISR



Hình 4.6 Lưu đồ giải thuật chương trình ngắt timer 1

4.7 Lưu đồ giải thuật chương trình con tính tần số – CALCULATE_HZ

- Các biến số **T_ON_3**, **T_ON_2** và **T_ON_1** lần lượt là byte 2, byte 1 và byte 0 của biến để lưu số chu kì máy (MC) của mức cao (1) trong một chu kì sóng vuông
- Biến **COUNT** là biến đếm số lượng chu kì xung vuông qua chân P3.2 trong thời gian 1 giây (đây cũng chính là tần số) $F = \frac{1}{T} = \frac{1}{1/COUNT} = COUNT$



Hình 4.7 Lưu đồ giải thuật chương trình con tính tần số

4.8 Lưu đồ giải thuật tính Dulty Cycle – CALCULATE_DUTY_CYCLE

- Với cách tính **Dulty Cycle** thì em chọn cách tính như sau:

$$DC = \frac{Số\ lượng\ chu\ kì\ trong\ 1\ giây \times số\ chu\ kì\ máy\ của\ mức\ 1\ trong\ 1\ chu\ kì}{10^4}\ \%$$

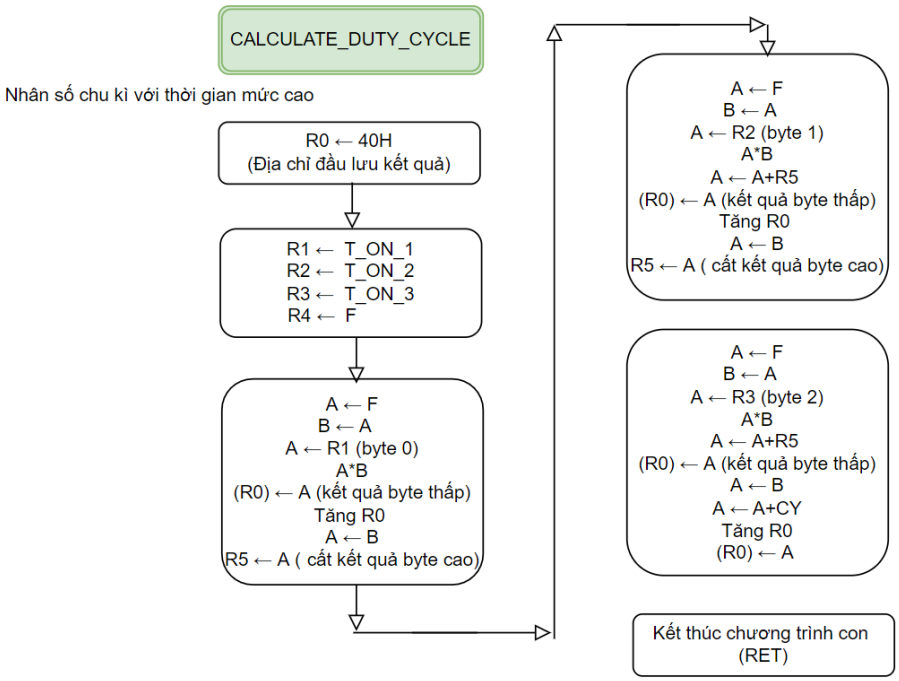
- Chương trình con tính Duty Cycle có nhiệm vụ tính **phép nhân** trên tử số của công thức phía trên:

  Số lượng chu kì trong 1 giây* × *Số chu kì máy của mức 1 trong 1 giây

  Trong đó:

  1. *Số lượng chu kì trong 1 giây* (tần số) được lưu trong biến **F**
  2. *Số chu kì máy của mức 1 trong 1 chu kì* (3 byte) được lưu lần lượt trong **T_ON_3 T_ON_2 T_ON_1**
  3. Kết quả của phép nhân (4 byte) được lưu trong bộ nhớ nội có địa chỉ lần lượt là 43H 42H 41H 40H

- Còn việc chia cho 10^4 cũng khá đơn giản khi chúng ta chuyển dấu chấm thập phân sang trái 4 digits trong lúc xuất kết quả ra màn hình LCD (công việc này sẽ được thực hiện trong chương trình con LCD_DISPLAY)

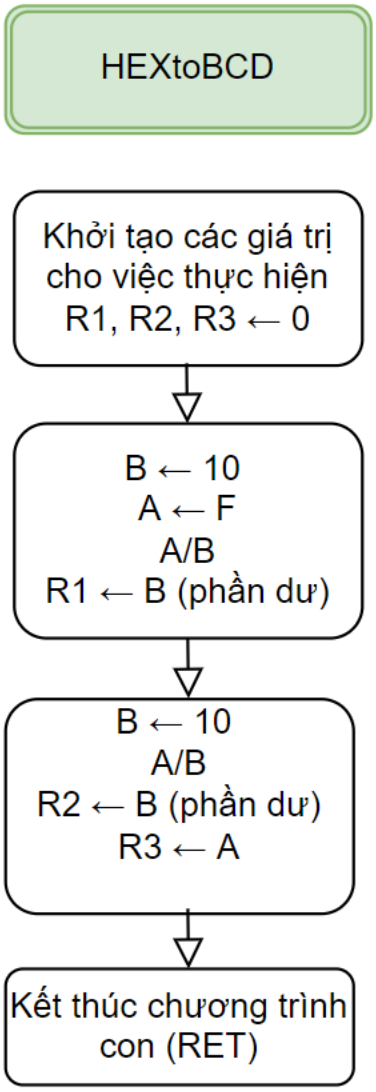


Hình 4.8 Lưu đồ giải thuật chương trình con tính duty cycle

4.9 Lưu đồ giải thuật chương trình chuyển HEX 8 bit sang ASCII 7 digits

Đây là chương trình con để chuyển giá trị của tần số (trong biến **F)** sang giá trị BCD để thuận tiện cho việc xuất kết quả ra màn hình

Kết quả thu được lưu trong **R3 R2 R1** tương ứng với byte từ cao đến thấp

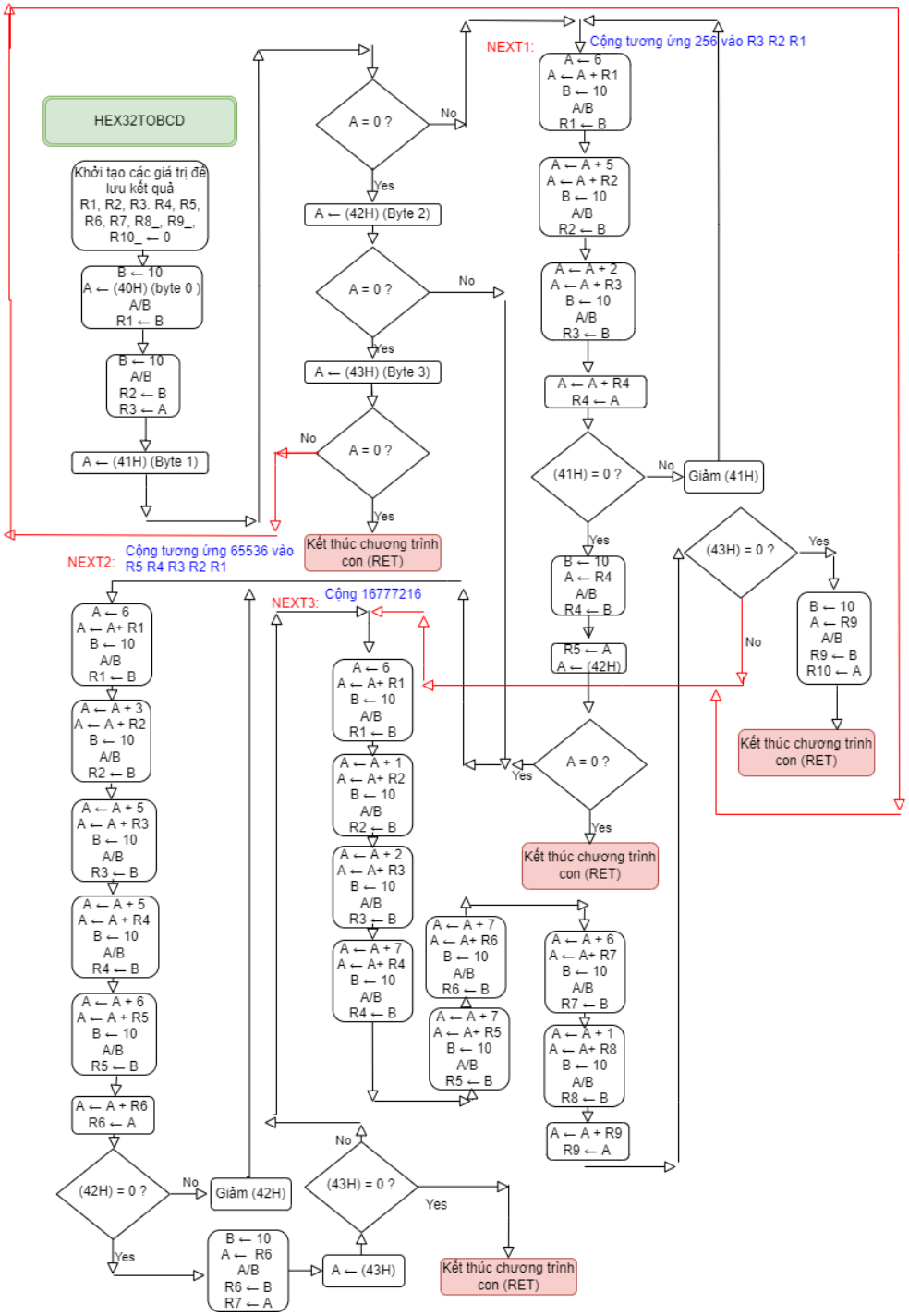


Hình 4.9 Lưu đồ giải thuật chương trình con chuyển HEX 8 bit sang BCD

4.10 Lưu đồ giải thuật chuyển HEX 32 bit sang ASCII 7 digits

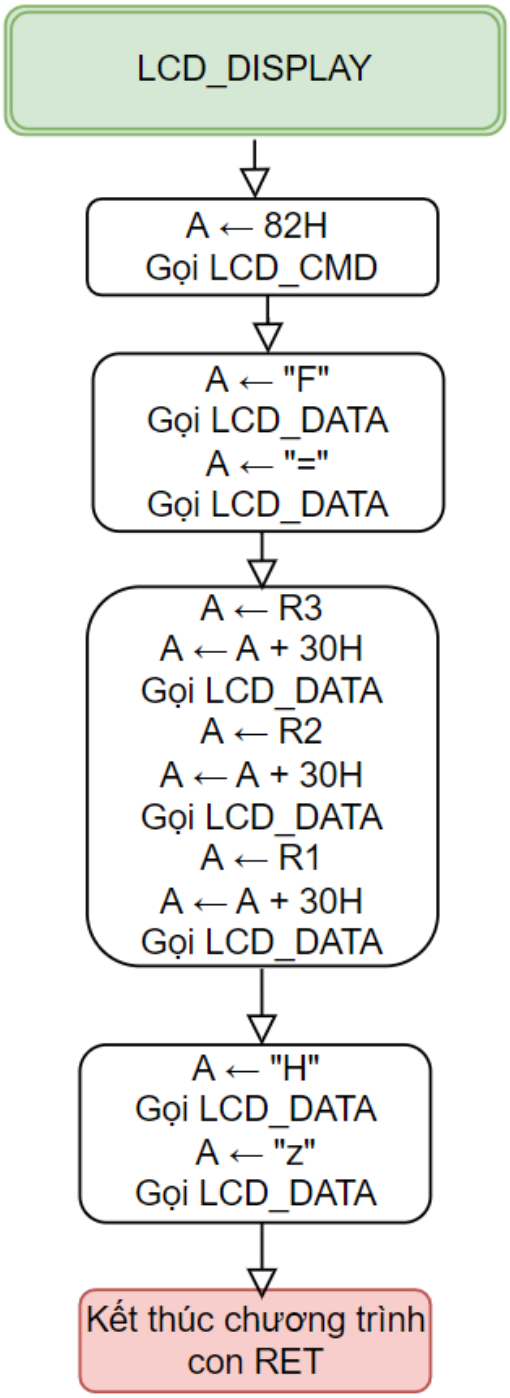
Chương trình con này dùng để chuyển giá trị kết quả của chương trình con CALCULATE_DUTY_CYCLE (tính theo MC**)** sang giá trị BCD để thuận tiện cho việc xuất kết quả ra màn hình

Kết quả thu được lưu trong **R10_ R9_ R8_ R7 R6 R5 R4 R3 R2 R1** tương ứng với byte từ cao đến thấp



Hình 4.10 Lưu đồ giải thuật chương trình con HEX 32 bit sang BCD

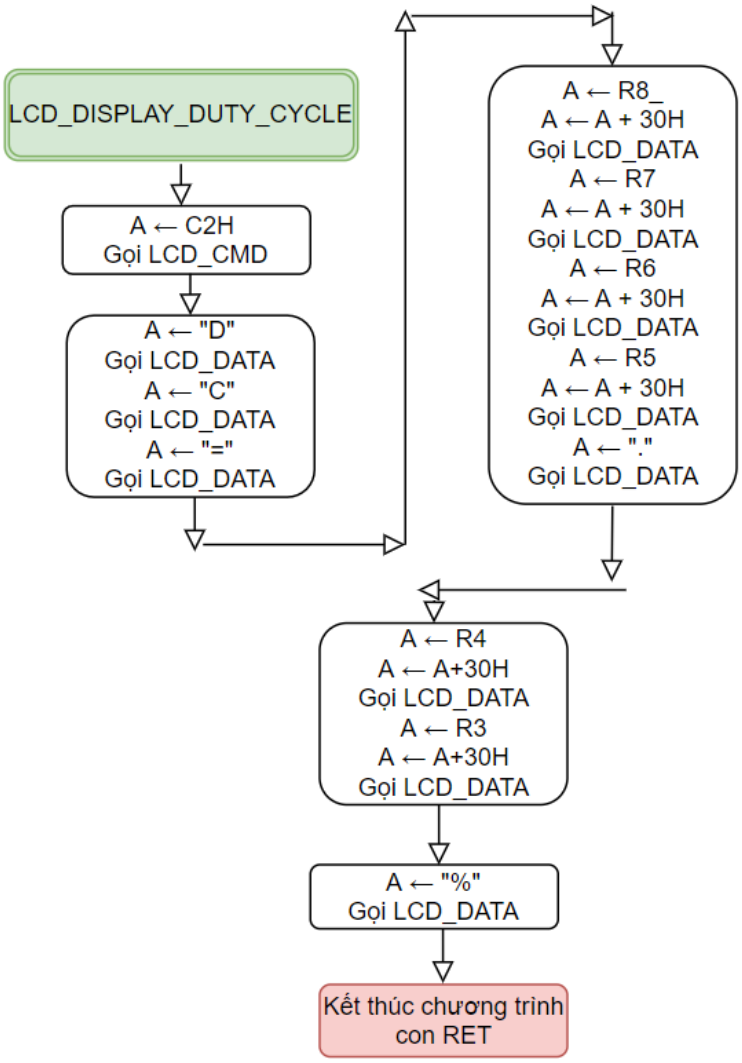
4.11 Lưu đồ giải thuật hiển thị tần số ra hàng 1 LCD – LCD_DISPLAY



Hình 4.11 Lưu đồ giải thuật hiển thị tần số ra hàng 1 LCD

4.12 Lưu đồ giải thuật hiển thị duty cycle ra hàng 2 LCD

LCD_DISPLAY_DUTY_CYCLE

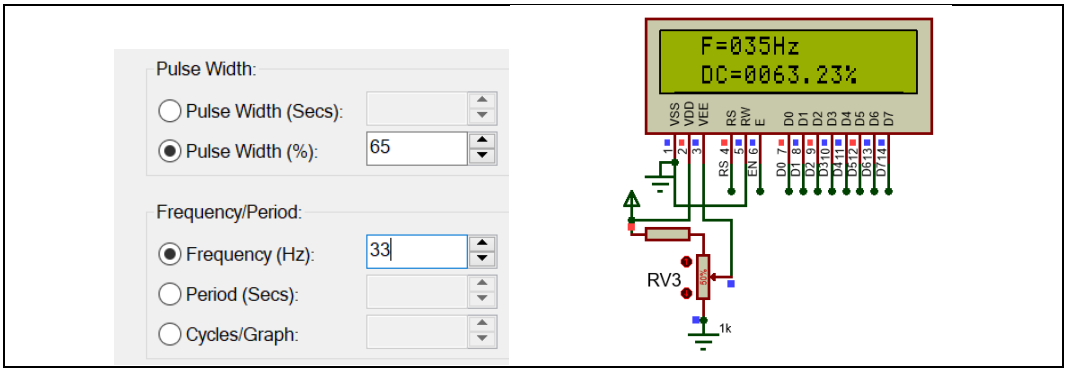


Hình 4.12 Lưu đồ giải thuật chương trình con hiển thị duty cycle ra hàng 2 LCD

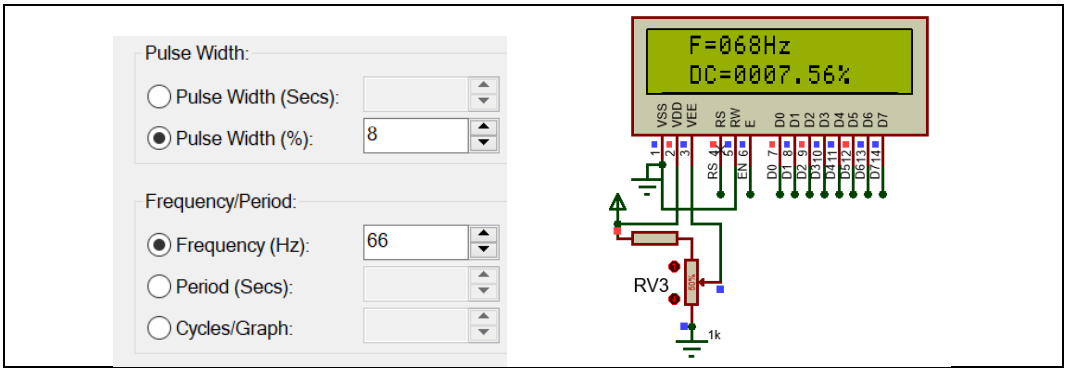
5. KẾT QUẢ MÔ PHỎNG BẰNG PROTEUS

Mô phỏng một số trường hợp với phần mềm Proteus:

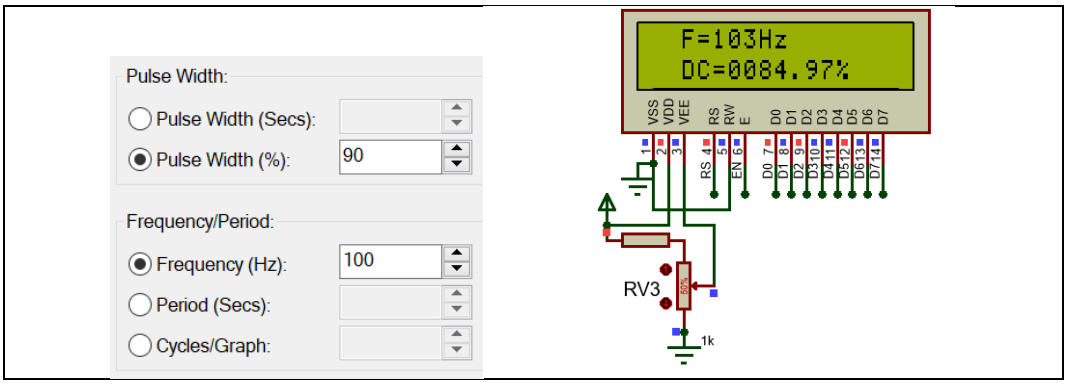
- $F = 33\ Hz$ và $DC = 65\ \%$



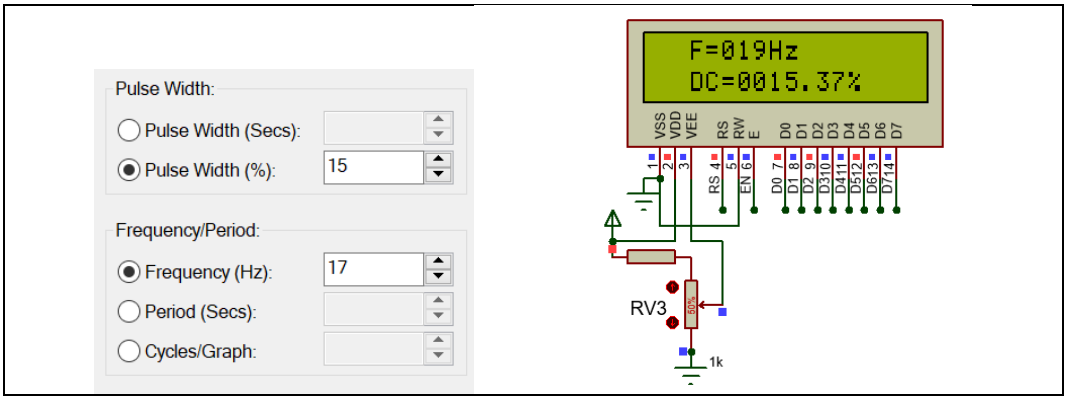
- $F = 66\ Hz$ và $DC = 8\ \%$



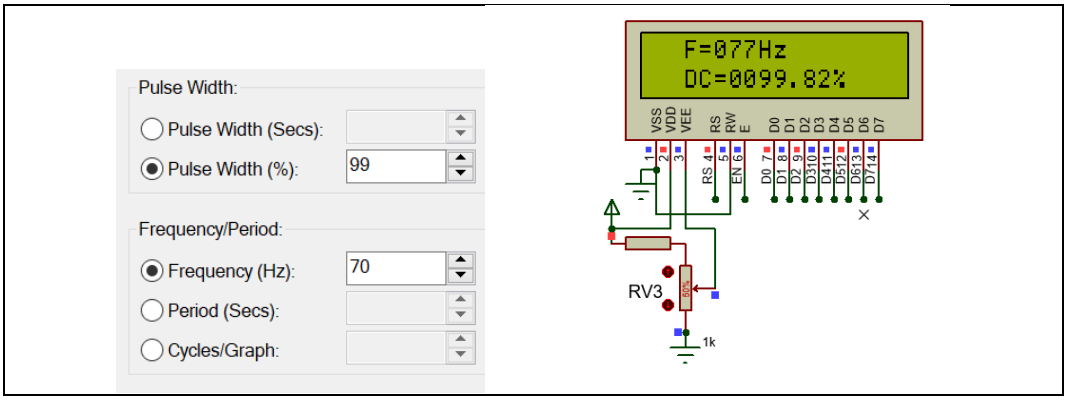
- $F = 100\ Hz$ và $DC = 90\ \%$



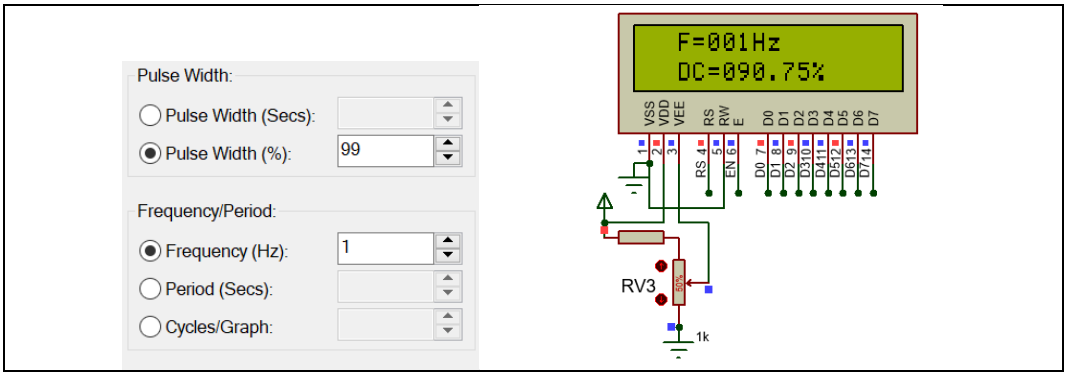
- $F = 17\ Hz$ và $DC = 15\%$



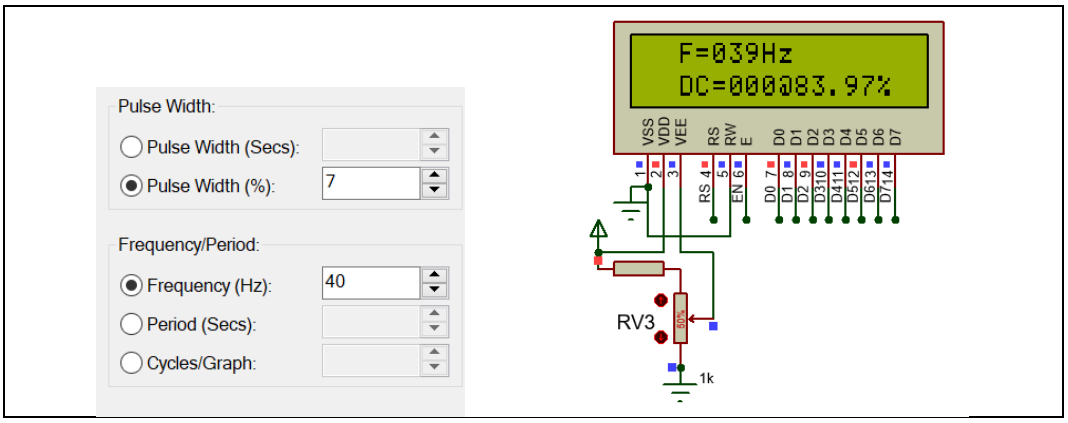
- $F = 70\ Hz$ và $DC = 99\%$



- $F = 1\ Hz$ và $DC = 99\%$



- **Tuy nhiên vẫn có một số trường hợp mô phỏng vẫn còn bị lỗi phần duty cycle: đa số là những trường hợp có DC rất nhỏ. Ví dụ như trường hợp: $F = 40\ Hz$ và $DC = 7\%$**



Bảng số liệu thu được:

STT	Lí thuyết		Mô phỏng		% Sai số	
	Tần số (Hz)	DC (%)	Tần số (Hz)	DC (%)	Tần số (Hz)	DC (%)
1	33	65	35	63.33	6.06 %	2.57%
2	66	8	68	7.58	3.03%	5.25%
3	100	90	103	84.97	3%	5.59%
4	17	15	19	15.37	11.77%	2.47%
5	70	99	77	99.82	10%	0.83%
6	1	99	1	90.75	0%	8.33%

% Sai số trung bình	
Tần số (Hz)	DC (%)
5.64%	4.17%

6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1 Kết luận

Trong quá trình thực hiện đề tài, em đã:

- Củng cố lại các kiến thức về kết nối phần cứng, LCD và tập lệnh của vi điều khiển 8051
- Hiểu rõ hơn về hoạt động ngắt và sử dụng các timer/counter
- Tiếp cận với các lưu đồ giải thuật giúp nâng cao khả năng tư duy và hiểu vấn đề sâu hơn
- Tham khảo, tổng hợp và lựa chọn các phương pháp thích hợp với yêu cầu của đề tài

Sau khi hoàn thành đề tài thì em rút ra những ưu điểm, nhược điểm như sau:

Ưu điểm:

- Mạch có cấu tạo khá đơn giản bao gồm những linh kiện, IC quen thuộc và giá thành hợp lý
- Dễ dàng sử dụng, tiện lợi
- Mạch hiển thị các kết quả lên LCD nên người dùng dễ dàng theo dõi tần số\

Nhược điểm

- Còn có sai số về giá trị đo được (khoảng 5%) do sai số về linh kiện, sai số về thời gian căn chỉnh các timer và sai số trong khi tính toán thiết kế
- Mạch chưa có đầy đủ các khối chức năng để có thể triển khai trong thực tế như khối nguồn tạo điện áp

6.2 Hướng phát triển

Mạch ứng dụng này có thể phát triển để có thể đo được khoảng tần số lớn hơn bằng cách sử dùng nhiều byte hơn để chứa giá trị tần số thay vì chỉ dùng 1 byte của biến COUNT như trong bài này

Điều chỉnh các khoảng Delay và Timer để giảm thiểu sai số giúp hệ thống chính xác hơn, đáng tin cậy hơn

Xử lí các vấn đề hiển thị trên LCD. Mở rộng menu lựa chọn và hướng dẫn sử dụng chi tiết để người dùng dễ dàng tiếp cận

Có thể mở rộng hệ thống ra để đo tần số lưới điện công nghiệp với khoảng đo 45-55 Hz

7. TÀI LIỆU THAM KHẢO

[1] Circuit Digest, "8051 Microcontroller based Frequency Counter", https://circuitdigest.com/microcontroller-projects/8051-microcontroller-based-frequency-counter

[2] Rickey World, "Frequency Counter using AT89C2051 and LCD (Assembly)", https://www.8051projects.net/download-d221-frequency-counter-using-at89c2051-lcd-assembly.html

[3] Embedds, "Very simple frequency counter on 8051 Microcontroller", https://embedds.com/very-simple-frequency-counter-on-8051-microcontroller/

[4] Archive – The Beginner's Arsenal, "Frequency Measurement using 8051 Microcontroller", https://archive.thebearsenal.com/2016/01/frequency-measurement-using-8051.html

[5] Study Microcontroller, "Frequency counter using 8051 microcontrooler|Assembly language program", https://www.youtube.com/watch?v=hPdkifWL41c

[6] Craig Hollinger, "Implementing a Frequency Counter Using a Microcontroller", https://www.youtube.com/watch?v=LNPMQlQPJTM

[7] Okashtein, "Digital Frequency meter for 8051", https://okashtein.wordpress.com/2013/04/13/digital-frequency-meter-for-8051-microcontroller-using-assembly/

8. PHỤ LỤC

Mã nguồn Assembly của hệ thống

```
;-------------------------------------------------------------------------------------------
------------#!
;---------------------------------{{Tran Minh Nhat}}-----------------------------------
--------#!
;----------------------{{Digital Frequency Meter}}-----------------------------------
#!
;--------------------------{{System: 8051 microcontroller}}---------------------------
------------#!
;-----------------------------{{Released at: 9/6/2022}}-----------------------------
-----------#!
;-------------------------------------------------------------------------------------------
------------#!
;khai bao ket noi phan cung;
            EN          BIT   P2.7        ;#!chan P2.7 la EN cua LCD
            RS          BIT   P2.5        ;#!chan P2.5 la RS cua LCD
            LCD         EQU   P1          ;#!khai bao port du lieu LCD
            F           EQU   30H         ;#!khai bao bien tan so F
            T_ON_1      EQU   31H         ;#!chua du lieu thoi gian muc cao
            T_ON_2      EQU   32H
            T_ON_3      EQU   33H
            R8_         EQU   50H         ;#!thanh ghi tao them cho phep tinh
            R9_         EQU   51H         ;#!thanh ghi tao them cho phep tinh
            R10_        EQU   52H         ;#!thanh ghi tao them cho phep tinh
            COUNT       EQU   53H         ;#!bien dem de tinh tan so
;-------------------------------------------------------------------------------------------
------------#!
;-----------------------------------------------------------------#!
;                     ~ CHUONG TRINH MAIN  ~
;-----------------------------------------------------------------#!
;1.Su dung ngat ngoai 0 de dem tan so va ngat timer 1 mode 1 (16bit) de tinh time_on,
;
;2.Bo dem se reset sau moi 1 giay dung timer 0, mode 1, timer 16 bit
;
;3.Hien thi thong bao len man hinh LCD
;
;-----------------------------------------------------------------#!
            ORG   0000H
            AJMP MAIN
            ORG   0003H                   ;#!vector ngat ngoai 0
            AJMP  EX0_ISR
            ORG   001BH                   ;#!vector ngat timer 1
            AJMP  ET1_ISR

            ORG   0030H
MAIN:       ACALL LCD_INIT                ;#!khoi tao man hinh LCD
AGAIN:      ACALL CALCULATE_HZ            ;#!tinh tan so
            ACALL HEXtoBCD                ;#!chuyen doi so hex sang BCD de hien thi
            ACALL LCD_DISPLAY             ;#!hien thi tan so ra man hinh LCD
            ;
            ACALL CALCULATE_DUTY_CYCLE    ;#!tinh chu ki nhiem vu duty cycle
            ACALL HEX32TOBCD              ;#!chuyen doi so hex 32 bit sang BCD hien thi
            ACALL LCD_DISPLAY_DUTY_CYCLE  ;#!hien thi duty cycle ra man hinh LCD
            AJMP  AGAIN                   ;#!lap vong tro lai
;--------------------------------------
EX0_ISR:
            INC   COUNT
            RETI
ET1_ISR:
            INC   T_ON_3
            RETI
```

```
;------------------------------------------------------------#!
;                     ~ chuong trinh con LCD_INIT ~
;------------------------------------------------------------#!
               ;khoi tao LCD 16x2
               ;gui cac ma code LCD de khoi tao 38H,0CH,01H,06H,8BH
               ;dung phuong phap tra bang
;------------------------------------------------------------#!
LCD_INIT:
            MOV   DPTR, #CODE_INIT              ;#!dat dia chi tra bang
LOOP1:      CLR   A                             ;#!xoa co ACC cho lenh tiep theo
            MOVC  A, @A+DPTR                    ;#!tra bang
            ACALL LCD_CMD                       ;#!goi chuong trinh con ghi lenh ra LCD
            INC   DPTR                          ;#!tang gia tri DPTR  <DPTR = DPTR + 1>
            JNZ   LOOP1                         ;#!nhay vong lai neu ACC khac 0
            RET
              RET
       ;#!return if zero to MAIN routine
;-------------------------------------------------------------------#!
;                     ~ chuong trinh con CALCULATE_HZ ~
;-------------------------------------------------------------------#!
               ;tinh so chu ki trong 1 giay
               ;
               ;dung timer 0 delay 1 giay
               ;cho timer0 chay 16 vong voi gia tri 62500
               ; 16 x 62500 = 1 000 000 uSec = 1 Sec
               ;
               ;dem so chu ki dung bien COUNT
;-------------------------------------------------------------------#!

CALCULATE_HZ:
            MOV   TMOD, #00010001B        ;#!timer 1 & 0 mode 1 (16bit)
            SETB  EA                      ;#!cho phep ngat toan cuc
            SETB  IT0                     ;#!bat canh xuong cua /INT0
            ;
            MOV   TL1, #00H               ;#!khoi tao cac gia tri bien tinh
            MOV   TH1, #00H
            MOV   T_ON_1, #00H
            MOV   T_ON_2, #00H
            MOV   T_ON_3, #00H
            MOV   COUNT, #00H

            ;#!tinh thoi gian muc 1 (5V)

            JB    P3.2, $
            JNB   P3.2, $                 ;#!cho canh len
            SETB  TR1                     ;#!cho timer1 chay de do thoi gian time_on
            SETB  ET1                     ;#!cho phep ngat timer1
            JB    P3.2, $                 ;#!doi muc 1
            CLR   TR1                     ;#!sau khi muc 1 ketthuc thi dung timer1
            MOV   T_ON_1, TL1             ;#!tra ve gia tri dem duoc vao T_ON_1 va
T_ON_2
            MOV   T_ON_2, TH1
            CLR   ET1                     ;#!cam ngat timer1

            ;#!do tan so

            SETB  EX0                     ;#!cho phep ngat ngoai 0
            MOV   R7, #15                 ;#!delay 1 Sec
BACK:       MOV   TL0, #LOW(-62500)
            MOV   TH0, #HIGH(-62500)
            SETB  TR0
            JNB       TF0, $
            CLR       TR0                     ;#!dung timer
            CLR       TF0                     ;#!xoa co bao tran TF0
            DJNZ  R7, BACK
```

```
            MOV   F, COUNT                ;#!tra gia tri tan so ve F
            CLR   EX0                     ;#!cam ngat ngoai 0
            RET


;-------------------------------------------------------------------#!
;                     ~ chuong trinh con CALCULATE_DUTY_CYCLE ~
;-------------------------------------------------------------------#!
               ;tinh chu ki nhiem vu
               ;do thoi gian time_on trong mot chu ki
               ;DC = time_on x tanso
;-------------------------------------------------------------------#!
CALCULATE_DUTY_CYCLE:
            ;#! thuc hien phep nhan time_on voi tanso

            MOV R0, #40H            ;#!dia chi dau luu ket qua
            MOV R1, T_ON_1          ;#!gan
            MOV R2, T_ON_2
            MOV R3, T_ON_3
            MOV R4, F
            MOV A, R4
            MOV B, A
            MOV A, R1
            MUL AB
            MOV @R0, A
            INC R0
            MOV A, B
            MOV R5, A
            MOV A, F
            MOV B, A
            MOV A, R2
            MUL AB
            ADD A, R5
            MOV @R0, A
            INC R0
            MOV A, B
            MOV R5, A
            MOV A, F
            MOV B, A
            MOV A, R3
            MUL AB
            ADD A, R5
            MOV @R0, A
            MOV A, B
            ADDC A, #00H
            INC R0
            MOV @R0, A
            RET
;-------------------------------------------------------------#!
;                     ~ chuong trinh con HEXtoBCD ~
;-------------------------------------------------------------#!
       ;chuyen so HEX 8 bit sang ASCII 7 digits
       ;ngo vao data byte = F
       ;ngo ra hang tram = R3
       ;ngo ra hang chuc = R2
       ;ngo ra hang don vi = R1
;-------------------------------------------------------------#!

HEXtoBCD:
            MOV       R1,#00H               ;#!don vi
            MOV       R2,#00H               ;#!chuc
            MOV       R3,#00H               ;#!tram
            ;
            MOV   B, #10
            MOV   A, F
            DIV   AB
```

```
            MOV   R1, B
            ;
            MOV   B, #10
            DIV   AB
            MOV   R2, B
            MOV   R3, A
            RET
;-------------------------------------------------------------------------#!
;                     ~ chuong trinh con HEX32TOBCD ~
;-----------------------------------------------------------------#!
       ;chuyen so hex 32 bit sang ASCII 7 digits
       ;input Data byte 3 = (43H)
       ;input Data byte 2 = (42H)
       ;input Data byte 1 = (41H)
       ;input Data byte 0 = (40H)

;-----------------------------------------------------------------#!
HEX32TOBCD:
            MOV R1, #00H            ;#!khoi tao gia tri ban dau cho cac bien
            MOV R2, #00H
            MOV R3, #00H
            MOV R4, #00H
            MOV R5, #00H
            MOV R6, #00H
            MOV R7, #00H
            MOV R8_, #00H
            MOV R9_, #00H
            MOV R10_, #00H
            ;
            MOV B, #10
            MOV A, 40H
            DIV AB
            MOV R1, B
            ;
            MOV B, #10
            DIV AB
            MOV R2, B
            MOV R3, A
            ;
            MOV A, 41H
            CJNE A, #0H, NEXT1_
            MOV A, 42H
            CJNE A, #0H, NEXT2_
            MOV A, 43H
            CJNE A, #0H, NEXT3_
            RET
            ;
NEXT1_:                             ;#!cong 256 tuong ung vao R1, R2, R3
            MOV A, #6
            ADD A, R1
            MOV B, #10
            DIV AB
            MOV R1, B
            ;
            ADD A, #5
            ADD A, R2
            MOV B, #10
            DIV AB
            MOV R2, B
            ;
            ADD A, #2
            ADD A, R3
            MOV B, #10
            DIV AB
            MOV R3, B
            ;
```

```
            ADD A, R4
            MOV R4, A
            DJNZ 41H, NEXT1_
            MOV B, #10
            MOV A, R4
            DIV AB
            MOV R4, B
            ;
            MOV R5, A
            MOV A, #42H
            CJNE A, #0H, NEXT2_
            RET
            ;
NEXT2_:                             ;#!cong 65536 tuong ung vao R1, R2, R3, R4, R5
            MOV A, #6
            ADD A, R1
            MOV B, #10
            DIV AB
            MOV R1, B
            ;
            ADD A, #3
            ADD A, R2
            MOV B, #10
            DIV AB
            MOV R2, B
            ;
            ADD A, #5
            ADD A, R3
            MOV B, #10
            DIV AB
            MOV R3, B
            ;
            ADD A, #5
            ADD A, R4
            MOV B, #10
            DIV AB
            MOV R4, B
            ;
            ADD A, #6
            ADD A, R5
            MOV B, #10
            DIV AB
            MOV R5, B
            ;
            ADD A, R6
            MOV R6, A
            DJNZ 42H, NEXT2_
            MOV B, #10
            MOV A, R6
            DIV AB
            MOV R6, B
            MOV R7, A
            ;
            MOV A, 43H
            CJNE A, #0H, NEXT3_
            RET
            ;
NEXT3_:                  ;#!cong 16777216 tuong ung vao R1, R2, R3, R4, R5, R6, R7, R8
            MOV A, #6
            ADD A, R1
            MOV B, #10
            DIV AB
            MOV R1, B
            ;
            ADD A, #1
            ADD A, R2
```

```
            MOV B, #10
            DIV AB
            MOV R2, B
            ;
            ADD A, #2
            ADD A, R3
            MOV B, #10
            DIV AB
            MOV R3, B
            ;
            ADD A, #7
            ADD A, R4
            MOV B, #10
            DIV AB
            MOV R4, B
            ;
            ADD A, #7
            ADD A, R5
            MOV B, #10
            DIV AB
            MOV R5, B
            ;
            ADD A, #7
            ADD A, R6
            MOV B, #10
            DIV AB
            MOV R6, B
            ;
            ADD A, #6
            ADD A, R7
            MOV B, #10
            DIV AB
            MOV R7, B
            ;
            ADD A, #1
            ADD A, R8_
            MOV B, #10
            DIV AB
            MOV R8_, B
            ;
            ADD A, R9_
            MOV R9_, A
            DJNZ 43H, NEXT3_
            MOV B, #10
            MOV A, R9_
            DIV AB
            MOV R9_, B
            MOV R10_, A
            RET

;-------------------------------------------------------------#!
;                    ~ chuong trinh con LCD_DISPLAY ~
;-------------------------------------------------------------#!
;             ;In du lieu tan so ra man hinh LCD
;-------------------------------------------------------------#!
LCD_DISPLAY:
            MOV A, #82H
            ACALL LCD_CMD
            ;
            MOV A, #"F"
            ACALL LCD_DATA
            ;
            MOV A, #"="
            ACALL LCD_DATA
            ;
            MOV A, R3
```

```
            ADD A, #30H
            ACALL LCD_DATA
            ;
            MOV A, R2
            ADD A, #30H
            ACALL LCD_DATA
            ;
            MOV A, R1
            ADD A, #30H
            ACALL LCD_DATA
            ;
            MOV A, #"H"
            ACALL LCD_DATA
            MOV A, #"z"
            ACALL LCD_DATA
            RET
;-------------------------------------------------------------#!
;          ~ chuong trinh con LCD_DISPLAY_DUTY_CYCLE ~
;-------------------------------------------------------------#!
;                  ;In du lieu duty cycle ra man hinh LCD
;-------------------------------------------------------------#!
LCD_DISPLAY_DUTY_CYCLE:
            MOV A, #0C2H
            ACALL LCD_CMD
            ;
            MOV A, #"D"
            ACALL LCD_DATA
            ;
            MOV A, #"C"
            ACALL LCD_DATA
            ;
            MOV A, #"="
            ACALL LCD_DATA
            ;
            MOV A, R8_
            ADD A, #30H
            ACALL LCD_DATA
            ;
            MOV A, R7
            ADD A, #30H
            ACALL LCD_DATA
            ;
            MOV A, R6
            ADD A, #30H
            ACALL LCD_DATA
            ;
            MOV A, R5
            ADD A, #30H
            ACALL LCD_DATA
            ;
            MOV A, #"."
            ACALL LCD_DATA
            ;
            MOV A, R4
            ADD A, #30H
            ACALL LCD_DATA
            ;
            MOV A, R3
            ADD A, #30H
            ACALL LCD_DATA
            ;

            MOV A, #"%"
            ACALL LCD_DATA

            RET
```

```
;-------------------------------------------------------------#!
;                     ~ chuong trinh con LCD_CMD ~
;-------------------------------------------------------------#!
;              ;De gui lenh dieu khien ra cac port LCD
;-------------------------------------------------------------#!
LCD_CMD:
              MOV      LCD, A                      ;#!gui noi dung code sang P1
              CLR      RS                          ;#!RS=0 vao mode command
              SETB     EN                          ;#!E=1 tao xung len
              ACALL DELAY                          ;#!delay
              CLR      EN                          ;#!E=0 de tao canh xuong
              RET

;-------------------------------------------------------------#!
;                     ~ chuong trinh con LCD_DATA ~
;-------------------------------------------------------------#!
;              ;De gui noi dung hien thi ra cac port LCD
;-------------------------------------------------------------#!
LCD_DATA:
              MOV      LCD, A                      ;#!gui du lieu sang P1
              SETB     RS                          ;#!RS=1 vao mode data
              SETB     EN                          ;#!E=1 tao xung len
              ACALL DELAY                          ;#!delay
              CLR      EN                          ;#!E=0 de tao canh xuong
              RET
;-------------------------------------------------------------#!
;                     ~ chuong trinh con  DELAY ~
;-------------------------------------------------------------#!
;
;-------------------------------------------------------------#!
 DELAY:
              MOV      R0,#255
              DJNZ     R0,$
              RET
;-------------------------------------------------------------------------------------------
------------#!
;-------------------------------------------------------------------------------------------
------------#!
CODE_INIT:     DB    38H,0CH,01H,06H,8BH,0
              END
```