TRƯỜNG ĐẠI HỌC BÁCH KHOA ĐÀ NẴNG

KHOA ĐIỆN TỬ – VIỄN THÔNG

BÁO CÁO

THỰC TẬP CÔNG NHÂN

Máy phát tần số

GVHD : Thầy Lê Hồng Nam

SVTH : Tổ 01 – Nhóm 9A

Võ Ngọc Trí Minh	06DT1
Trương Thị Kim Ngà	06DT2
Sengsay Somsamay	06DT1
Trần Phước Thịnh	06DT1
Nguyễn Thị Bảo Trâm	06DT1
Nguyễn Lê Hoàng Tuấn	06DT1

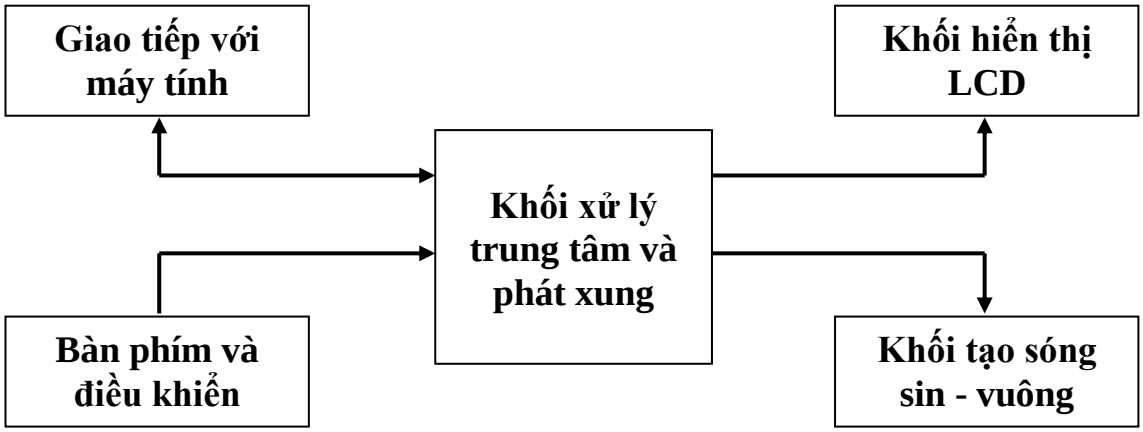
Đà Nẵng – 2010

MỤC LỤC

1. **Yêu cầu của đề tài:**

- Thiết kế mạch phát tần số từ 20Hz-1MHz, có thể hoạt động độc lập hoặc thực hiện giao tiếp với máy tính.

2. **Sơ đồ khối :**

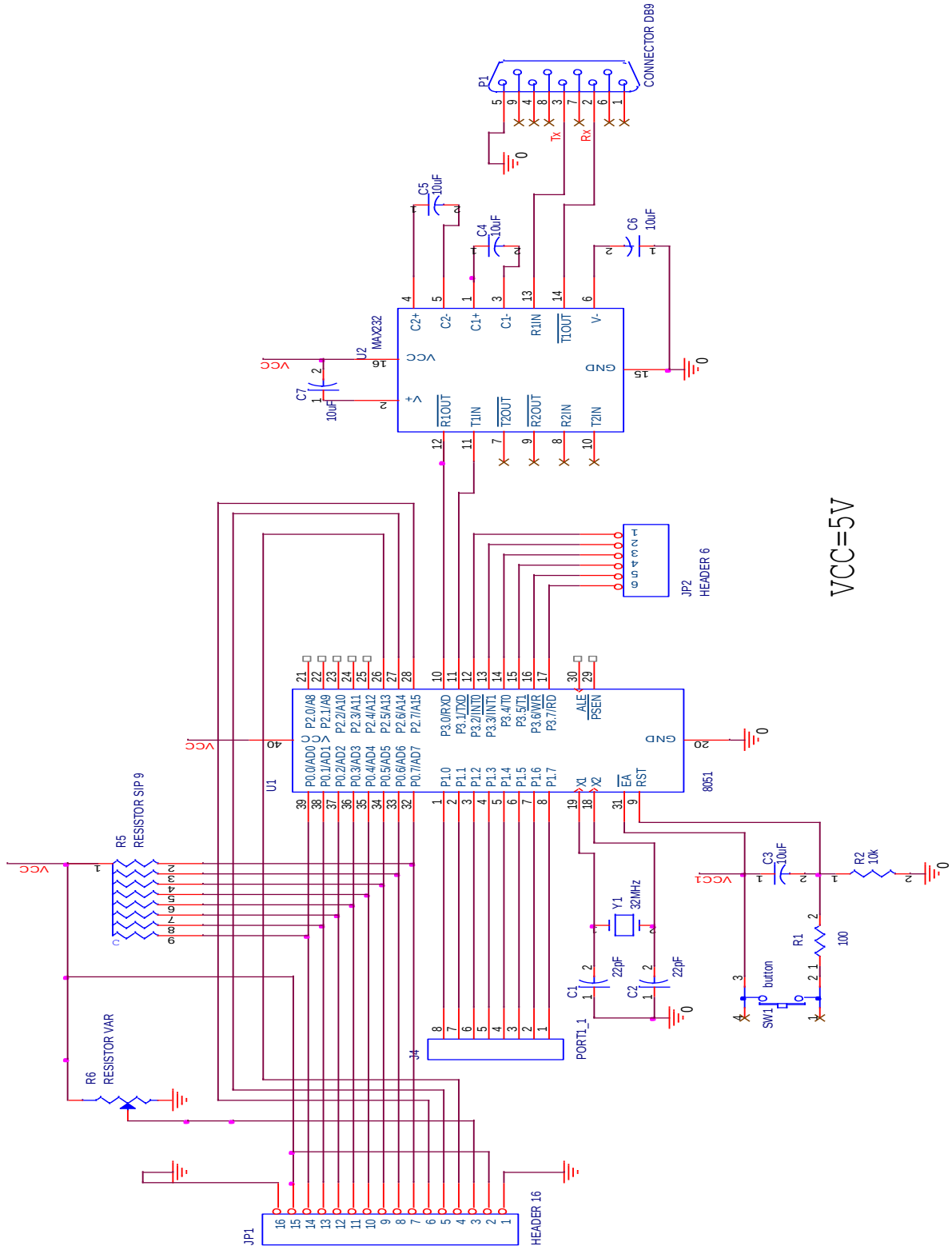


2.1 Khối xử lý trung tâm và phát xung :

a. **Nhiệm vụ :** Xử lý tín hiệu điều khiển, xử lý dữ liệu đầu vào của tần số, thực hiện chức năng giao tiếp với máy tính , hiển thị giá trị tần số lên LCD.

b. **Chọn linh kiện :**
   - ATMEL P89V51RD2
   - Các Tụ
   - Thạch Anh 24MHz
   - Điện trở thanh 1K
   - Header thanh

c. **Sơ đồ nguyên lý :**


CONNECTOR DB9
P1
MAX232
U2
C1
C2
C3
C4
C5
C6
C7
10uF
22pF
32MHz
Y1
R1
100
R2
10k
R5
RESISTOR SIP 9
R6
RESISTOR VAR
SW1
button
8051
U1
JP1
HEADER 16
JP2
HEADER 6
PORT1_1
VCC=5V

d. Nguyên lý làm việc của mạch :

- Nhận dữ liệu điều khiển từ bàn phím 4x4 qua port1.
- Giao tiếp với PC qua cổng Com dùng max232 bằng các chân RxD (P3.0) và TxD (P3.1).
- Chuyển giá trị tần số nhập sang mã ASCII gửi đến port 0 để hiển thị lên LCD.
- Tín hiệu xung vuông được phát ra ở chân P3.7

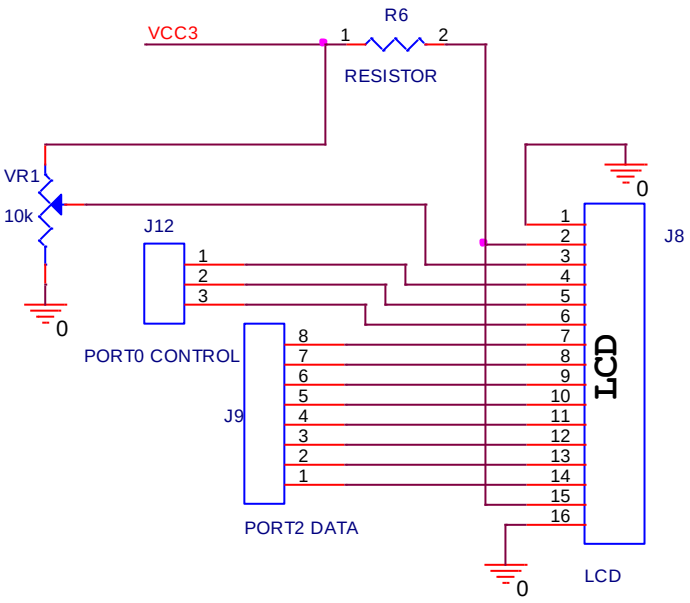
2.2 Khối hiển thị LCD :

a. Nhiệm vụ : khối này có nhiệm vụ hiển thị giá trị tần số phát được yêu cầu.

b. Chọn linh kiện :

- LCD 16X2 (3bit Control, 8bit Data)
- VR 10k để tăng giảm độ sáng đèn nền LCD

c. Sơ đồ nguyên lý :



d. Nguyên lý làm việc của mạch :

- Hoạt động trên chế độ 8 bit dữ liệu.
- Biến trở dùng để điều khiển độ sáng tối của LCD.
- LCD nhận lệnh từ VĐK để thiết lập trạng thái và nhận dữ liệu gửi xuống (đã được chuyển sang mã ASCII) thông qua port 0 để hiển thị.

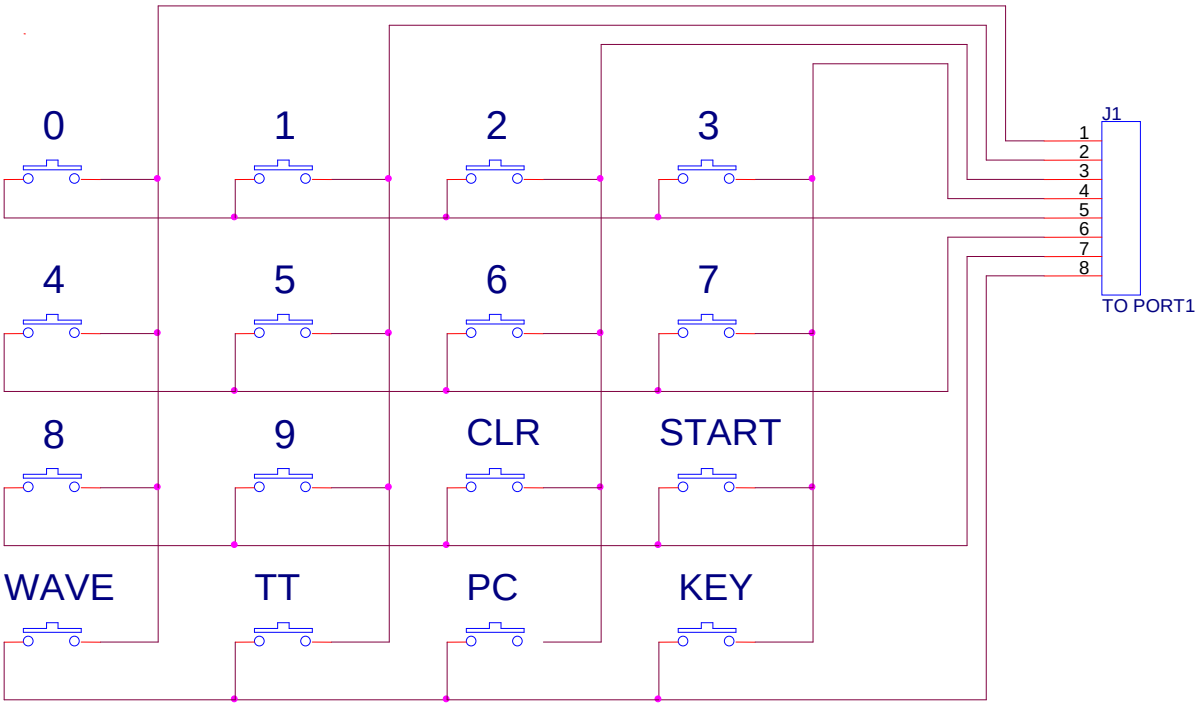
2.3 Bàn phím và điều khiển :

- **Nhiệm vụ :** sử dụng để nhập giá trị tần số cần phát và lựa chọn cách thức nhập giá trị, lựa chọn dạng sóng ngõ ra, đưa ra các yêu cầu xóa (CLEAR), bắt đầu (START) và tiếp tục (CONTINUE).

a. Chọn linh kiện :

- Switch (dạng nút bấm)
- Header thanh

b. Sơ đồ nguyên lý :



c. Nguyên lý làm việc của mạch :

- Dùng phương pháp quét phím. Quét cột kiểm tra hàng hoặc ngược lại. Các cột cách nhau 1 đơn vị, các hàng cách nhau 4 đơn vị.

- Giá trị của bàn phím được xác định :

  - Bp=C+4xH
  - Trong đó :Bp là giá trị của phím bấm
    C là cột được quét
    H là hàng có phím được bấm.

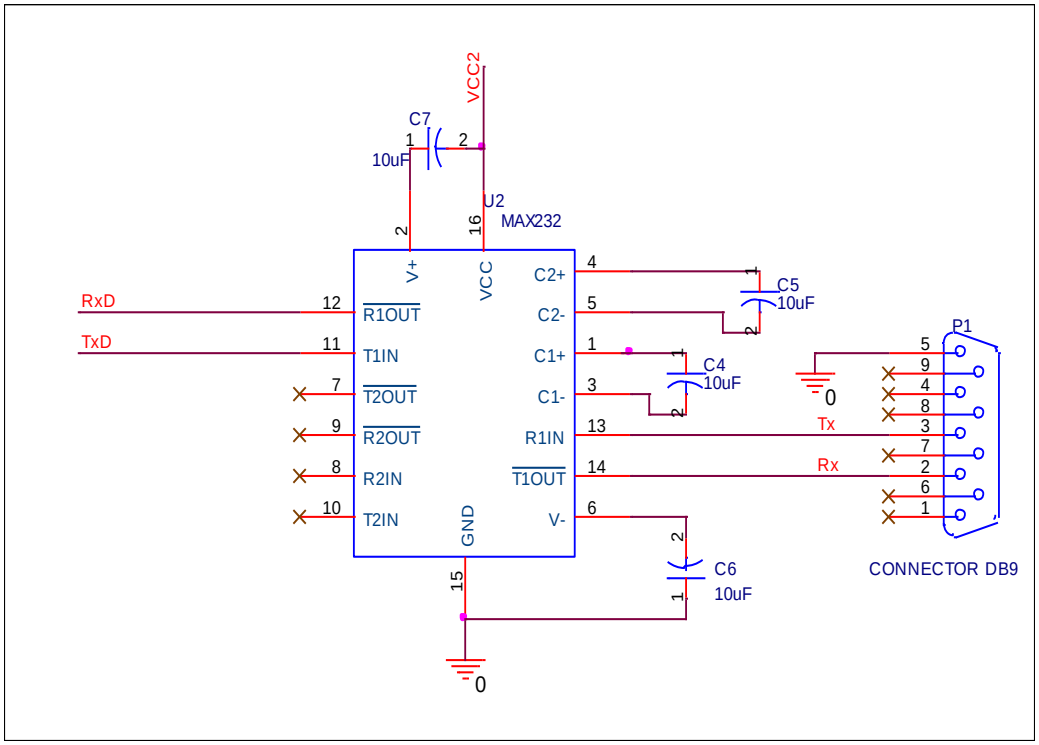
- Sau khi nhập tần số phát ta bấm phím WAVE để ra lệnh cho VĐK phát sóng sin, nếu không mặc định sẽ là phát xung vuông.
- Nếu dữ liệu nhập vào không đúng ta có thể bấm CLEAR để tiến hành nhập lại.
- Để bắt đầu phát xung ta bấm START.

2.4 Khối giao tiếp máy tính :

a. Nhiệm vụ : máy tính sẽ thực hiện việc gửi dữ liệu tần số phát xuống VĐK (nếu có) và sẽ giám sát hoạt động phát tần số của VĐK thông qua cổng COM theo chuẩn logic RS232.

b. Chọn linh kiện : MAX232 và cổng COM

c. Sơ đồ nguyên lý :



d. Nguyên lí làm việc của mạch :

- MAX232 sẽ chuyển mức logic từ TTL sang 232 v à ngược lại, thực hiện việc đồng bộ điện thế giữa PC và VĐK.

+Chuẩn TTL :

- Space :0.1 → 0.8V
- Mark :2 → 2.4V

+Chuẩn 232 :

- Space :3 → 25V
- Mark :-3 → -25V

- Dữ liệu từ máy tính sẽ được truyền nối tiếp vào VĐK thông qua khối giao tiếp máy tính, sau đó nhận thông tin phản hồi từ VĐK để thực hiện việc giám sát. VĐK dùng hai chân RxD và TxD để thực hiện việc giao tiếp.

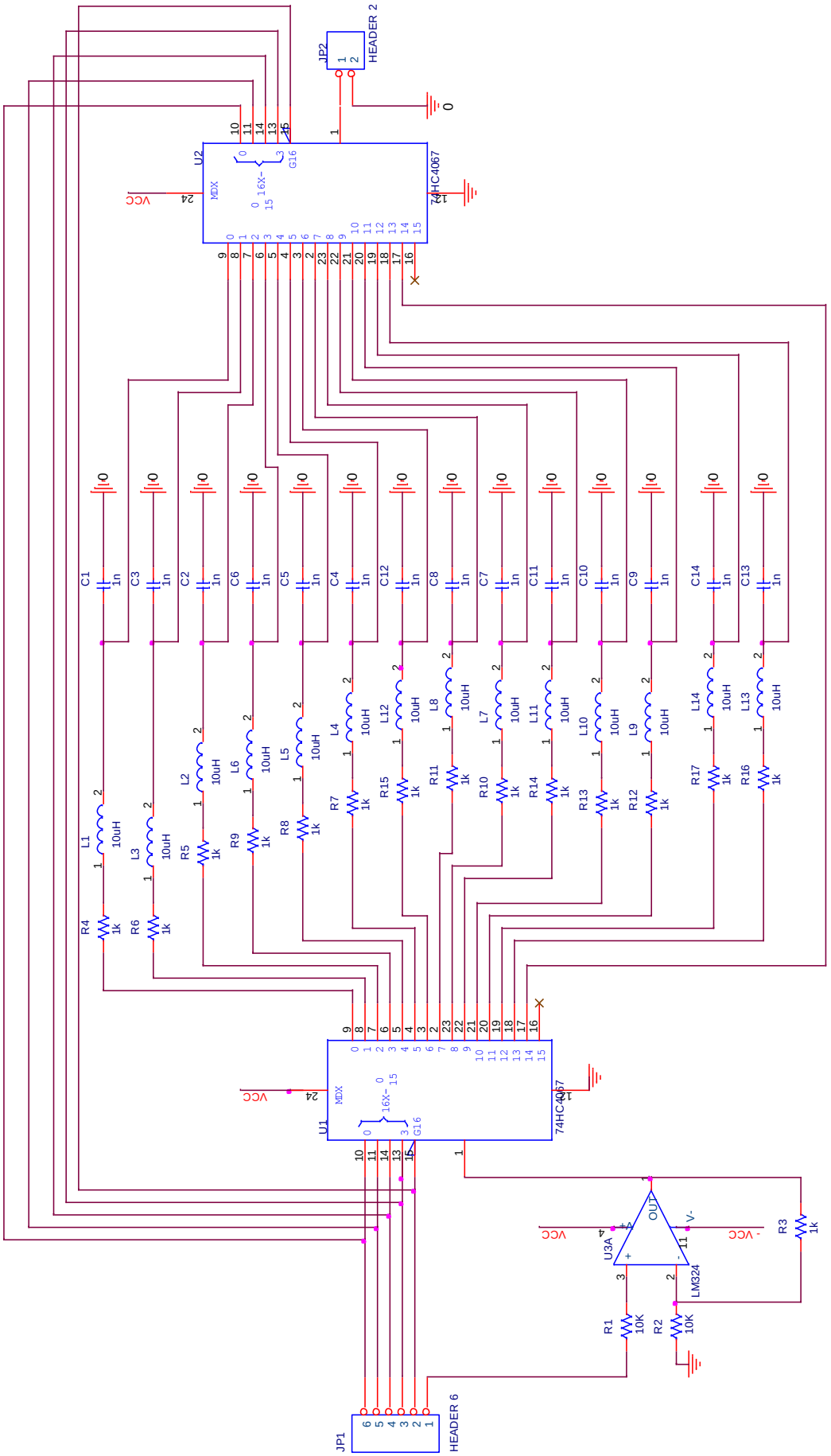
2.5 Khối tạo sóng sin :

a. Nhiệm vụ : tùy theo nhu cầu của ứng dụng mà mạch sẽ phát dạng sóng là sin hay vuông, yêu cầu này được điều khiển thông qua bàn phím hay giao tiếp với máy tính

b. Chọn linh kiện:

- IC 74HC4051, IC 74HC4067, IC LM324, cuộn cảm 100mH, các tụ điện và điện trở.

c. Sơ đồ nguyên lý :


JP2
HEADER 2
U2
74HC4067
U1
74HC4067
U3A
LM324
JP1
HEADER 6
VCC
0
R1 10K
R2 10K
R3 1k

d. Nguyên lí làm việc của mạch :

- Sóng vuông lấy ra từ chân P3.7 của VĐK sẽ được khuếch đại biên độ điện áp nhờ một mạch khuếch đại không đảo dùng opamp, tín hiệu sau khuếch đại sẽ được đưa vào một IC chuyển mạch ngõ vào 74HC4067. IC này hoạt động dựa trên 5 bit điều khiển (1 bit enable và 4 bit địa chỉ) được chỉ định bởi VĐK để đưa tín hiệu sau khuếch đại trực tiếp đến IC 74HC4067 lựa chọn ngõ ra thực hiện việc phát xung vuông ; hoặc sẽ đưa tín hiệu sau khuếch đại vào các bộ lọc thông thấp rồi mới đến IC chuyển mạch ngõ ra để thực hiện phát sin.

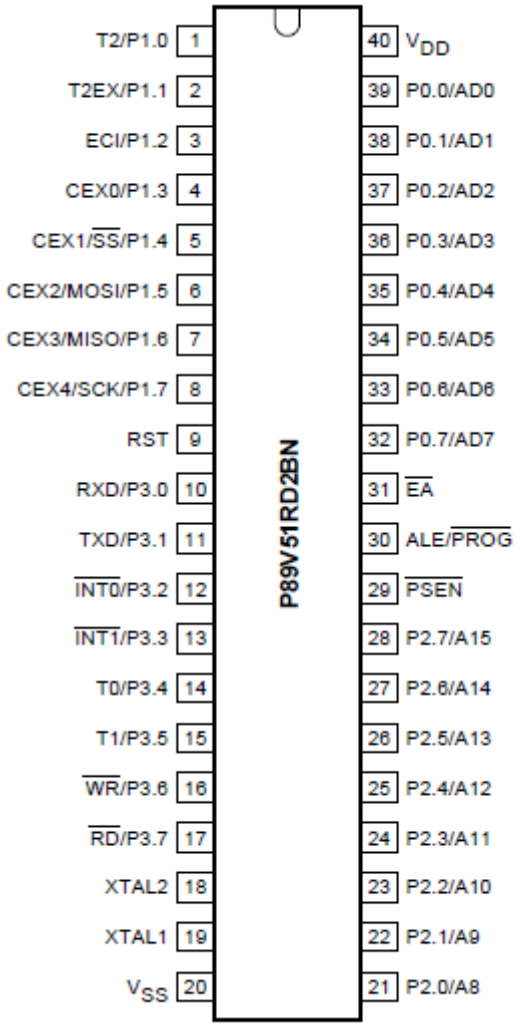
3. Giới thiệu các linh kiện sử dụng :

3.1 Vi điều khiển 89V51RD2 :

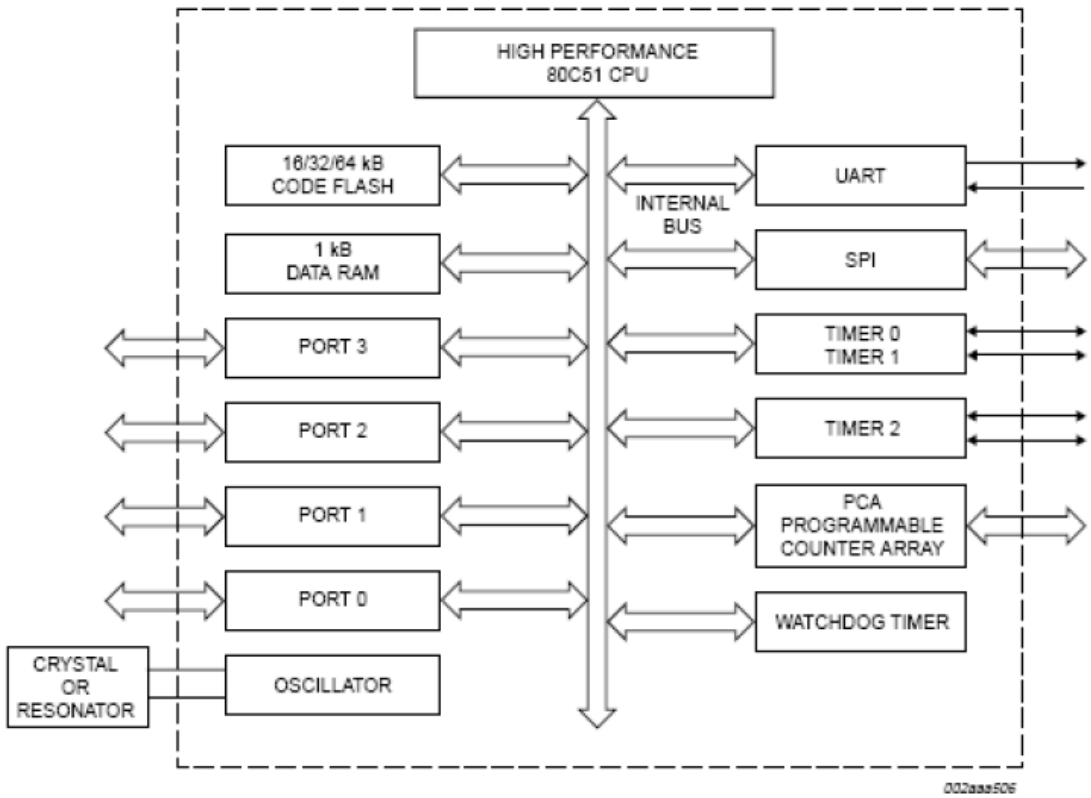
a. Khái quát các tính năng:

Trước tiên, ta lướt qua các tính năng của P89V51RD2:

- Khái quát:
  - P89V51RD2 là vi điều khiển 80C51 có 64kB Flash và 1024bytes<1kB> bộ nhớ dữ liệu RAM.
  - Tính năng đặc biệt của P89V61RD2 là ở chế độ hoạt động mode x2. Người thiết kế chọn chạy ứng dụng của mình ở chế độ này để nâng đôi tốc độ khi hoạt động ở cùng tần số dao động<một chu kì máy=6 chu kì xung nhịp>
  - Bộ nhớ chương trình Flash cho phép lập trình ISP hoặc/và song song. Chế độ lập trình song song được đưa ra để thích ứng với tốc độ cao, giảm thời gian và giá thành.
  - IAP/ISP.
- Các tính năng:
  - CPU 80C51.
  - Hoạt động ở 5VDC trong tầm tần số dao động đến 40MHz.
  - 64kB ISP.
  - SPI
  - 5 PCA với chức năng PWM/capture/compare 16bits.
  - 4 cổng xuất nhập.
  - 3 Timers/Couters 16bits.
  - Watchdog Timer có thể lập trình được.
  - 8 nguồn ngắt.
  - 2 thanh ghi DPTR.
  - Tương thích mức logic TTL và CMOS.
  - Phát hiện nguồn yếu <Brownout Detect>
  - Chế độ Low-power, Power down, Idle.
- Sơ đồ chân của MCU P89V51RD2:



- Sơ đồ khối của MCU P89V51RD2:



Hình 1: Sơ đồ khối của MCU P89V51RD2

Sơ qua về các chân của vi điều khiển:

- Port 0, Port 1, Port 2, Port 3: Như cấu trúc 8051 kinh điển.
- P1.0 - T2: Ngõ vào Counter cho Timer/Counter 2 hoặc ngõ ra cho Counter/Timer 2.
- P1.1 - T2EX: Điều khiển hướng và cạnh kích chức năng Capture cho timer/Counter 2.
- P1.2 – ECI: Ngõ vào xung nhịp. Tín hiệu này là nguồn xung nhịp ngoài cho chức năng PCA.
- P1.3 –CEX0: ngõ vào xung nhịp cho chức năng Capture/Compare modul 0.
- P1.4:
  - SS: Chọn cổng phụ vào cho SPI.
  - CEX1: ngõ vào xung nhịp cho chức năng Capture/Compare modul 1.
- P1.5:
  - MOSI: phục vụ SPI
  - CEX2: ngõ vào xung nhịp cho chức năng Capture/Compare modul 2.
- P1.6:
  - MISO: phục vụ SPI
  - CEX3: ngõ vào xung nhịp cho chức năng Capture/Compare modul 3.
- P1.7:
  - SCK: phục vụ SPI
  - CEX4: ngõ vào xung nhịp cho chức năng Capture/Compare modul 4.
- PSEN: Cho phép dùng bộ nhớ chương trình ngoài. Khi MCU sử dụng bộ nhớ chương trình trong chip, PSEN không tích cực. Khi sử dụng bộ nhớ chương trình ngoài, PSEN thường ở mức tích cực 2 lần trong mỗi chu kì máy. Sự chuyển mức cao sang thấp trên ↓PSEN cưỡng bức từ bên ngoài khi ngõ vào RST đang ở mức cao trong hơn 10 chu kì máy sẽ đưa MCU vào chế độ lập trình host từ bên ngoài.
- RST: Khi nguồn dao động đang hoạt động, mức cao trên chân RST trong ít nhất 2 chu kì máy sẽ Reset lại hệ thống. Nếu chân PSEN chuyển mức trong khi RST vẫn còn ở mức cao, MCU sẽ vào chế độ lập trình host từ bên ngoài, nếu không, sẽ hoạt động bình thường.
- EA: Cho phép sử dụng bộ nhớ chương trình ngoài.
  - EA=’0’:Bộ nhớ ngoài.
  - EA=’1’:Bộ nhớ trong chip.
- ALE/PROG: Cho phép khóa địa chỉ<Như 8051 cổ điển> ngoài ra, chân này còn được dùng để đưa vào chế độ lập trình FLASH.

b. Tổ chức bộ nhớ:

MCU P89V51RD2 có 2 vùng không gian địa chỉ riêng biệt: vùng lưu trữ cho bộ nhớ chương trình và cùng địa chỉ cho bộ nhớ dữ liệu <cấu trúc Harvard> .

Lựa chọn bank bộ nhớ chương trình flash:

- Có 2 vùng bộ nhớ nội flash trong MCU, Block 0 có 64kB và được tổ chức thành 512 sector, mỗi sector chứa 128 Bytes. Block 1 chứa chương trình ISP/ICP và được kích hoạt khi chọn kết hợp bit reset mềm (SWR) <FCF.1> và bit chọn bank (BSEL) <FCF.0>.
- Quá trình tuần tự sau khi nguồn được bật, chương trình boot sẽ tự động thực thi và cố gắng lấy tín hiệu autobaud từ máy chủ. Nếu không có quá trình này xảy ra trong vòng 400ms và bit .

3.2 LCD 16X2 :

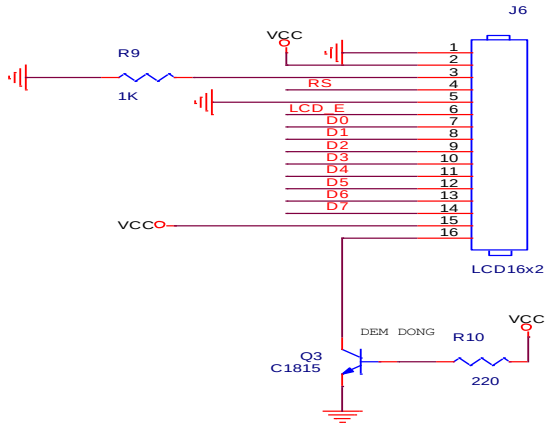
a. Hoạt động của LCD :

Trong những năm gần đây, màn hình tinh thể lỏng LCD (Liquid Crystal Display) ngày càng sử dụng rộng rãi nà đang dần thay thế các đèn LED (7 đoạn và nhiều đoạn). Đó là vì các nguyên nhân sau :

- Màn hình LCD có giá thành hạ.
- Khả năng hiển thị số, ký tự và đồ hoạ tốt hơn nhiều so với đèn LED (đèn LED chỉ hiển thị được số và một số ký tự).
- Sử dụng thêm một bộ điều khiển làm tươi LCD và như vậy giải phóng CPU khỏi công việc này. Còn đối với đèn LED luôn cần CPU (hoặc bằng cách nào đó) để duy trì việc hiển thị dữ liệu.
- Dễ dàng lập trình các ký tự và đồ hoạ.

b . Mô tả chân LCD:

- LCD giới thiệu ở đây có 14 chân. Chức năng của các chân được cho trong bảng 4.1.



LCD 16x2

Chân	Ký hiệu	I/O	Mô tả
1	V_{SS}	-	Đất
2	V_{CC}	-	Dương nguồn +5V
3	V_{EE}	-	Nguồn điều khiển tương phản
4	RS	I	RS = 0 chọn thanh ghi lệnh. RS = 1 chọn thanh ghi dữ liệu.
5	R/W	I	R/W = 1 đọc dữ liệu. R/W = 0 ghi.
6	E	I/O	Cho phép
7	DB0	I/O	Bus dữ liệu 8 bit
8	DB1	I/O	Bus dữ liệu 8 bit
9	DB2	I/O	Bus dữ liệu 8 bit
10	DB3	I/O	Bus dữ liệu 8 bit
11	DB4	I/O	Bus dữ liệu 8 bit

12	DB5	I/O	Bus dữ liệu 8 bit
13	DB6	I/O	Bus dữ liệu 8 bit
14	DB7	I/O	Bus dữ liệu 8 bit

V_{CC}, V_{SS} và V_{EE} : V_{CC} và V_{SS} là chân nguồn +5V và chân đất, còn V_{EE} được dùng để điều khiển độ tương phản của LCD.

RS (Register Select) - chọn thanh ghi : Có hai thanh ghi rất quan trọng bên trong LCD. Chân RS được dùng để chọn các thanh ghi này. Nếu RS = 0 thì thanh ghi mã lệnh được chọn, cho phép người dùng gửi một lệnh chẳng hạn như xoá màn hình, đưa con trỏ về đầu dòng v.v. Nếu RS = 1 thì thanh ghi dữ liệu được chọn và cho phép người dùng gửi dữ liệu cần hiển thị lên LCD.

R/W (Read/Write) – Chân đọc/ghi : Chân vào đọc/ghi cho phép người dùng đọc/ghi thông tin từ/lên LCD. R/W = 0 thì đọc, còn R/W = 1 thì ghi.

E (Enable) – Chân cho phép : Chân cho phép E được LCD sử dụng để chốt thông tin hiện có trên chân dữ liệu. Khi dữ liệu được cấp đến chân dữ liệu thì một xung mức Cao-xuống-Thấp được áp đến chân E để LCD chốt dữ liệu trên chân dữ liệu. Xung này phải rộng tối thiểu là 450ns.

D0 – D7 :

- Đây là 8 chân dữ liệu 8 bit, được dùng để gửi thông tin lên LCD hoặc đọc nội dung của các thanh ghi trong LCD.
- Để hiển thị chữ cái và con số, mã ASCII của các chữ cái từ A đến Z, a đến z và các con số từ 0 – 9 được gửi đến các chân này khi bật RS = 1.
- Cũng có các mã lệnh được gửi đến LCD để xoá màn hình hoặc đưa con trỏ về đầu dòng hoặc nhấp nháy con trỏ. Bảng 4.2 liệt kê các mã lệnh này.
- Cũng có thể sử dụng RS = 0 để kiểm tra bit cờ bận xem LCD đã sẵn sàng nhận thông tin chưa. Khi R/W = 1 và RS = 0 thì cờ bận D7 thực hiện các chức năng như sau : Nếu D7 = 1 (cờ bận bằng 1) có nghĩa LCD đang bận các công việc bên trong và sẽ không nhận bất kỳ thông tin mới nào, còn nếu D7 = 0 thì LCD sẵn sàng nhận thông tin mới. Trong mọi trường hợp cần kiểm tra cờ bận trước khi ghi bất kỳ dữ liệu nào lên LCD.

Mã lệnh LCD :

Mã (Hexa)	Lệnh đến thanh ghi của LCD
1	Xoá màn hình hiển thị
2	Trở về đầu dòng
4	Dịch con trỏ sang trái
6	Dịch con trỏ sang phải
5	Dịch hiển thị sang phải
7	Dịch hiển thị sang trái
8	Tắt con trỏ, tắt hiển thị
A	Tắt hiển thị, bật con trỏ

C	Bật hiển thị, tắt con trỏ
E	Bật hiển thị, nhấp nháy con trỏ
F	Tắt hiển thị, nhấp nháy con trỏ
10	Dịch vị trí con trỏ sang trái
14	Dịch vị trí con trỏ sang phải
18	Dịch toàn bộ hiển thị sang trái
1C	Dịch toàn bộ hiển thị sang phải
80	Đưa con trỏ về đầu dòng thứ nhất
C0	Đưa con trỏ về đầu dòng thứ hai
38	Hai dòng và ma trận 5 x 7
Ghi chú : Bảng này mở rộng từ bảng 4.4	

Gửi có trễ lệnh và dữ liệu đến LCD :

- Để gửi một lệnh bất kỳ nêu ở bảng 4.2 đến LCD, cần đưa chân RS = 0, còn để gửi dữ liệu thì bật RS = 1. Sau đó, gửi một sườn xung cao xuống thấp đến chân E để cho phép chốt dữ liệu trong LCD.
- Gửi mã lệnh hoặc dữ liệu đến LCD có kiểm tra cờ bận :Gửi có trễ lệnh và dữ liệu đến LCD như trên thì cần phải đặt một độ trễ lớn trong quá trình xuất dữ liệu hoặc lệnh ra LCD. Tuy nhiên, một cách tốt hơn nhiều là hiển thị cờ bận trước khi xuất một lệnh hoặc dữ liệu tới LCD.
- Lưu ý cờ bận D7 của thanh ghi lệnh ở chương trình. Để đọc thanh ghi lệnh cần đặt RS = 0, R/W = 1 và xung Cao-xuống-Thấp đến bit E. Sau khi đọc thanh ghi lệnh, nếu bit D7 (cờ bận) ở mức cao thì LCD bận và không có thông tin (lệnh) nào được xuất đến. Chỉ khi D7 = 0 mới có thể gửi dữ liệu hoặc lệnh đến LCD. Lưu ý trong phương pháp này không sử dụng trễ thời gian vì chúng ta đã kiểm tra cờ bận trước khi xuất lệnh hoặc dữ liệu lên LCD.
- Bảng dữ liệu của LCD :
- Ở LCD chúng ta có thể đặt dữ liệu vào bất cứ chỗ nào. Dưới đây là các địa chỉ và cách chúng truy cập.

RS	E/W	DB7	DB6	DB5	DB6	DB3	DB2	DB1	DB0
0	0	1	A	A	A	A	A	A	A

Trong đó, AAAAAAA = từ 0000000 đến 0100111 ở dòng lệnh 1 và AAAAAAA = 1000000 đến 1100111 ở dòng lệnh 2. Xem bảng 4.3.

Định địa chỉ cho LCD :

	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Dòng 1 (min)	1	0	0	0	0	0	0	0
Dòng 1 (max)	1	0	1	0	0	1	1	1
Dòng 2 (min)	1	1	0	0	0	0	0	0
Dòng 2 (max)	1	1	1	0	0	1	1	1

-Vùng địa chỉ cao có thể lên tới 0100111 đối với LCD 40 ký tự (0100111B=39), còn đối với LCD 20 ký tự thì chỉ đến 010011 (10011B=19D). Như vậy, đối với LCD 40 x 2 ta có địa chỉ theo thập phân là từ 0 đến 39.

-Từ những trình bày trên đây, chúng ta có thể xác định được địa chỉ của vị trí con trỏ đối với LCD có các kích thước khác nhau. Xem hình 4.16. Chú ý rằng, tất cả địa chỉ đều cho ở dạng Hexa. Hình 4.17 giới thiệu phân bố thời gian của LCD. Bảng 4.4 liệt kê chi tiết các lệnh của LCD.

Địa chỉ con trỏ của một số LED :

16 x 2 LCD	80	81	82	83	84	85	86	……	8F
	C0	C1	C2	C3	C4	C5	C6	……	CF
20 x 1 LCD	80	81	82	83	……	93			
20 x 2 LCD	80	81	82	83	……	93			
	C0	C1	C2	C3	…….	D3			
20 x 4 LCD	80	81	82	83	……	93			
	C0	C1	C2	C3	……	D3			
	94	95	96	97	……	A7			
	D4	D5	D6	D7	……	E7			
20 x 2 LCD	80	81	82	83	……	A7			
	C0	C1	C2	C3	……	E7			
Ghi chú : Tất cả dữ liệu đều ở dạng Hexa.									

3.3 Bàn phím 4X4 :

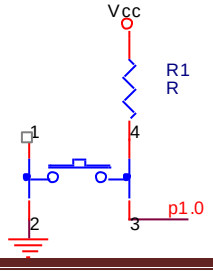
- Để thực hiện ma trận bàn phím ta dùng phương pháp quét phím.Quét cột và đọc dữ liệu tại hàng hoặc ngược lại. Theo hình vẽ thì các cột cách nhau 1 đơn vị, các hàng cách nhau 4 đơn vị.
- Vậy giá trị của bàn phím được tính theo công thức sau:

$$Bp = C + h.4$$

Trong đó: Bp: Giá trị của phím được nhấn.
C: Cột được quét.
H: Hàng có phím nhấn.

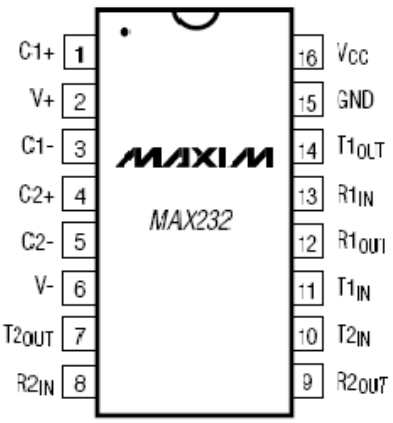
Ví dụ: Khi ta quét cột C0 mà phím 4 được nhấn thì H1 nhận được tín hiệu. Vậy giá trị nhận được của bàn phím là Bp = 0 + 1.4 = 4.

- Khi mạch cần nhiều phím thì ta mới tổ chức ma trận phím để giảm số lượng cổng sử dụng cho bàn phím.
- Khi mạch không cần nhiều phím thì ta có thể dùng phím như sau:

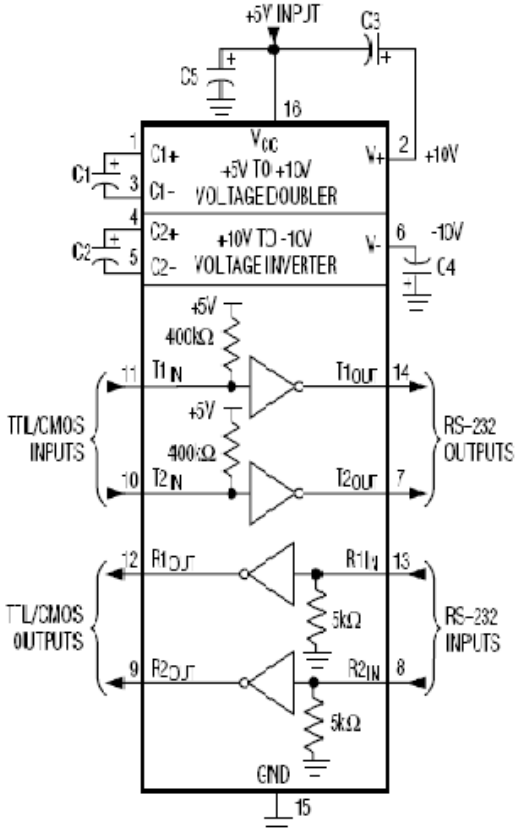


3.4 MAX 232 :

Sơ đồ chân :



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX232	1.0	1.0	1.0	1.0	1.0



Sơ đồ chân và mạch đặc trưng của Max232

- Các ứng dụng của max232:
  - Máy tính xách tay
  - Modem công suất thấp
  - Hệ thống ắc qui- năng lượng RS 232
  - Mạng đa điểm RS 232
- Các thông số kĩ thuật của max232:
  - Nguồn cung cấp: +5V.
  - Đặc trưng: tốc độ chuyển đổi cao hơn, đỉnh nhỏ.
  - Giá trị thông thường của tụ là: 1µF.
  - Data rate : 200kbps.

4. Sơ đồ nguyên lí tổng: *(mạch kèm theo)*

5. Giải thích sơ đồ nguyên lí làm việc của mạch:

- Khi mới cắm nguồn LCD hiển thị lời giới thiệu nhóm, sau đó sẽ hiển thị "f(Hz)=" ở dòng đầu tiên.
- VĐK sẽ dùng chương trình để kiểm tra phương thức nhập là từ máy tính hay bàn phím. Nếu máy tính được chọn thì việc nhập dữ liệu tần số cần phát và giám sát hoạt động của VĐK sẽ do máy tính nắm quyền điều hành. Nếu bàn phím được chọn thì việc nhập dữ liệu tần số cần phát sẽ do bàn phím thực hiện và việc giám sát hoạt động của VĐK vẫn do máy nắm quyền.
- Khi dữ liệu tần số được nhập thì LCD sẽ hiển thị tần số cần phát. Sau đó, ta sẽ lựa chọn phát sin hay vuông. Nếu tần số cần nhập đã đúng thì ta bắt đầu phát xung, nếu chưa đúng có thể xóa và nhập lại.
- Dạng sóng cơ bản phát ra trên chân P3.7 là xung vuông. Xung này sẽ được đưa qua khối tạo sóng sin vuông để tạo ra dạng sóng theo đúng yêu cầu.

6. Tính chọn linh kiện cho mạch:

6.1 Phần nguồn:

- -Để đảm bảo điện áp cung cấp cho các khối trong mạch, chúng ta sử dụng nguồn đôi 12V
- Khi đó điện áp tại ngõ ra của cầu diode:

$$V_{ra} = 12\sqrt{2} - 2 \times 0.7 = 15.57V$$

- Để điện áp DC ở ngõ ra tương đối bằng phẳng ta sử dụng tụ lọc 4700nF

6.2 Khối xử lý trung tâm và phát xung:

- Chọn trở thanh kéo lên nguồn ở port 0: Vì dòng ngõ ra tại port 0 rất thấp nên ta phải kéo trở lên nguồn để nâng dòng.Ta chọn trở thanh 1KΩ
- Chọn giá trị tần số của bộ dao động thạch anh:

Theo yêu cầu thiết kế tần số tối đa của xung là 1MHz.

Chu kỳ của xung ra:

$$T = \frac{1}{f_{xung}} = \frac{1}{1MHz} = 1\mu s$$

Độ rộng xung:

$$\tau = \frac{T}{2} = 0.5\mu s$$

Khi đó thời gian của một chu kỳ lệnh tối đa phải là 0.5 μs.Tần số thạch anh cần dùng phải là:

$$f_{dd} \geq \frac{12}{\tau} = \frac{12}{0.5} = 24MHz$$

Vậy ta chọn thạch anh 24MHz.

Chọn C_1=C_2=33pF

- Tính toán bộ switch cho chân RST của vi điều khiển.
  - + Khi dòng đổ qua trở R_1 và R_2,với mục đích muốn sụt áp chủ yếu rơi trên R_2 để quyết định mức điện áp cho chân RST nên R_2 phải lớn hơn nhiều so với R_1.
  - + Ta chọn R_1=100 Ω và R_2=10 KΩ
  - + Ta có phương trình xả của tụ C_3 từ 5Vdc xuống còn 2.6Vdc(lúc này áp rơi trên R_2 bằng 2.2Vdc-2.4Vdc nên RST ở mức cao):

$V_{C3}(t)=[V_{C3}(0)-V_{C3}(\infty)].e^{\frac{-t}{\tau}}+V_{C3}(\infty)$

$2.6 = [0-5].e^{\frac{-t}{\tau}}+5$

$\rightarrow e^{\frac{t}{\tau}} \approx 2.08$

$\frac{t}{\tau} \approx 0.734$

  - + Gọi t_{max} là thời gian lâu nhất cần thiết để bấm chân RST.Vì vậy, t_{max} càng bé càng tốt.Tuy nhiên,t_{max} phải phù hợp với thực tế,nên ta chọn t_{max}=0.1s.

$\rightarrow \tau=(R_1+R_2).C_3 \approx 0.136s$

$\rightarrow C_3=\frac{\tau}{(R1+R2)} = \frac{0.136}{100+10.10^3} \approx 13.47.10^{-6}F$

  - + Ta chọn C3=10μF

6.3 Khối bàn phím.

6.4 Khối giao tiếp máy tính.

6.5 Khối hiển thị LCD

6.6 Khối chuyển mạch chọn dạng tín hiệu ngõ ra.

a. Khối khuếch tín hiệu ngõ vào:

- Ta sử dụng một mạch khuếch đại không đảo dùng opamp.
- Vì điện áp nguồn cung cấp cho opamp lần lượt là +15V, -15V; đồng thời ngõ ra mức cao của vi điều khiển thuộc khối mạch trung tâm đạt gần 5V, nên ta phải tính toán thiết kế mạch khuếch đại tín hiệu vào lên gấp hai lần để tránh trường hợp điện áp tín hiệu ngõ ra vượt quá ngưỡng 15V.
- Ta có hệ số khuếch đại của mạch:

$$A_V = \left(1 + \frac{R_f}{R_{in}}\right) = 2$$

Với $R_{in} = 10\ KOhm \rightarrow R_f = R_{in} = 10\ KOhm$

b. Khối chuyển mạch:

- Vì tần số xung vuông ngõ vào thay đổi nên để chuyển sang dạng sóng sin ta phải cho qua nhiều bộ lọc vi phân bậc 2. Ta chia dãy tần số từ 20Hz đến 1MHz ra làm nhiều băng, trong mỗi băng ta chọn một tần số trung tâm để tính giá trị của C và R với L được chọn cố định ban đầu, ứng với mỗi cặp giá trị R-C tính ra thì mọi giá trị tần số trong băng này đều đáp ứng cho ra dạng sóng hình sin ít bị méo dạng nhất (có thể chấp nhận được).
- Với mỗi băng ta chọn tần số $f_0 = \frac{f_{max} + f_{min}}{2} = \frac{1}{2\pi\sqrt{LC}}$

  (thực ra trong dãy tần số từ f_{min} đến f_{max} có thể chọn bất kỳ tần số nào làm f_o)
- Hệ số phẩm chất của mạch: $Q = \frac{1}{R} \times \sqrt{\frac{L}{C}}$
- Để đặc tuyến của bộ lọc thông thấp (R,L,C) gần với đồ thị tiệm cận của nó thì hệ số phẩm chất của mạch phải nằm trong khoảng $\frac{1}{\sqrt{2}} < Q < 1$

  hay $\sqrt{\frac{L}{C}} < R = R_t + R_{ON-IC} + r_L < \sqrt{\frac{2L}{C}}$

  Ta có $R_{ON-IC} = 50\ Ohm, với\ V_{DD} của\ IC\ là\ 15V$,và nội trở của cuộn dây r_L=70 Ω

  Nên $\sqrt{\frac{L}{C}} - 120 < R_t < \sqrt{\frac{2L}{C}} - 120$
- Để phần lớn các hài của tín hiệu được truyền qua thì $f \leq f_H$(với f_H là tần số cắt trên.Trong thiết kế này ta chọn f_H=f_{max} của băng được chọn).Sự biến dạng của tín hiệu ở đầu ra tại các điểm gián đoạn là do các hài bậc cao $n > \frac{f_H}{f}$ đã bị loại đi. Để tạo ra tín hiệu hình sin ít bị méo dạng ta sẽ loại đi các hài có bậc lớn hơn 2.Điều này có nghĩa với thiết kế này chúng ta phải chọn tần số phải thỏa mãn $\frac{f_H}{f} \leq 2$.
- Trên cơ sở đó chúng ta có thể chọn băng theo điều kiện $\frac{f_{max}}{f_{min}} = 2$
- Như vậy từ 20Hz đến 1MHz ta sẽ chia được thành 16 băng như sau:

20Hz-40Hz	5.12KHz-10.24KHz
40Hz-80Hz	10.24KHz-20.48KHz
80Hz-160Hz	20.48KHz-40.96KHz
160Hz-320Hz	40.96KHz-81.92KHz
320Hz-640Hz	81.92KHz-163.84KHz
640Hz-1.28KHz	163.84KHz-327.68KHz
1.28KHz-2.56KHz	327.68KHz-655.36KHz
2.56KHz-5.12KHz	655.36KHz-1MHz

- Chọn cuộn cảm có giá trị 100mH (70 Ω)
  + ***Băng 20Hz-40Hz:***

$$f_{0-1} = \frac{40+20}{2} = 30Hz$$

$$C_{15} = \frac{1}{(2\pi f_{0-1})^2 \times L_1} = \frac{10}{(2\pi \times 30)^2} = 0.281 \times 10^{-3}F$$

Ta chọn :$C_{15} = 220\mu F$
Như vậy: $-98.67\ Ohm < R < -89.84\ Ohm$
Ta chọn: R_{15}=1 Ω
Điều này có nghĩa tín hiệu sin ở ngõ ra có biên độ thấp hơn đường tiệm cận của nó.Hay biên độ sóng sin thấp hơn so với biên độ xung vuông ngõ vào.

  + ***Băng 40Hz-80Hz:***

$$f_{0-2} = \frac{80+40}{2} = 60Hz$$

$$C_{16} = \frac{1}{(2\pi f_{0-2})^2 \times L_1} = \frac{10}{(2\pi \times 60)^2} = 7.03 \times 10^{-5}F$$

Ta chọn: $C_{16} = 100\mu F$
Như vậy: $-88.38\ Ohm < R_{16} < -75.28\ Ohm$
Ta chọn : R_{16}=1 Ω

  + ***Băng 80Hz-160Hz***:

$$f_{0-3} = \frac{160+80}{2} = 120Hz$$

$$C_{17} = \frac{1}{(2\pi f_{0-3})^2 \times L_1} = \frac{10}{(2\pi \times 120)^2} = 1.76 \times 10^{-5}F$$

Ta chọn: $C_{17} = 22\mu F$
Như vậy: $53.58\ Ohm < R_{17} < -24.66\ Ohm$
Ta chọn: R_{17}=1 Ω

  + ***Băng 160Hz-320Hz:***

$$f_{0-4} = \frac{320+160}{2} = 240Hz$$

$$C_{18} = \frac{1}{(2\pi f_{0-4})^2 \times L_1} = \frac{10}{(2\pi \times 240)^2} = 4.4 \times 10^{-6}F$$

Ta chọn: $C_{18} = 4.7\mu F$
Như vậy: $25.86\ Ohm < R_{18} < 86.28\ Ohm$
Ta chọn:R_{18}=27 Ω

  + ***Băng 320Hz-640Hz:***

$$f_{0-5} = \frac{640+320}{2} = 480Hz$$

$$C_{19} = \frac{1}{(2\pi f_{0-5})^2 \times L_1} = \frac{10}{(2\pi \times 480)^2} = 1.1 \times 10^{-6}F$$

Ta chọn: $C_{19} = 1\mu F$
Như vậy: $196.23\ Ohm < R_{19} < 327.21\ Ohm$
Ta chọn: R_{19}=220 Ω

+ ***Băng 640Hz-1.28KHz:***

$$f_{0-6} = \frac{640 + 1280}{2} = 960Hz$$

$$C_{20} = \frac{1}{(2\pi f_{0-6})^2 \times L_1} = \frac{10}{(2\pi \times 960)^2} = 2.74 \times 10^{-7} F$$

Ta chọn: $C_{20} = 0.22\mu F$
Như vậy: $554.2\ Ohm < R_{20} < 833.46\ Ohm$
Ta chọn: R_{20}=560 Ω

+ ***Băng 1.28KHz-2.56KHz:***

$$f_{0-7} = \frac{2.56 + 1.28}{2} = 1.92KHz$$

$$C_{21} = \frac{1}{(2\pi f_{0-7})^2 \times L_1} = \frac{10}{(2\pi \times 1.92 \times 10^3)^2} = 6.87 \times 10^{-8} F$$

Ta chọn: $C_{21} = 0.068\mu F$
Như vậy: $1.092\ KOhm < R_{21} < 1.594\ KOhm$
Ta chọn: R_{21}=1.2 KΩ

+ ***Băng 2.56Khz-5.12Khz:***

$$f_{0-8} = \frac{5.12 + 2.56}{2} = 3.84KHz$$

$$C_{22} = \frac{1}{(2\pi f_{0-8})^2 \times L_1} = \frac{10}{(2\pi \times 3.84 \times 10^3)^2} = 1.72 \times 10^{-8} F$$

Ta chọn: $C_{22} = 0.015\ \mu F$
Như vậy: $2.46\ KOhm < R_{22} < 3.53\ KOhm$
Ta chọn: R_{22}=2.7 KΩ

+ ***Băng 5.12KHz-10.24KHz:***

$$f_{0-9} = \frac{10.24 + 5.12}{2} = 7.68KHz$$

$$C_{23} = \frac{1}{(2\pi f_{0-9})^2 \times L_1} = \frac{10}{(2\pi \times 7.68 \times 10^3)^2} = 4.29 \times 10^{-9} F$$

Ta chọn: $C_{23} = 0.0047\mu F$
Như vậy: $4.49\ KOhm < R_{23} < 6.4\ KOhm$
Ta chọn: R_{23}=4.7 KΩ

+ ***Băng 10.24KHz-20.48KHz:***

$$f_{0-10} = \frac{20.48 + 10.24}{2} = 15.36KHz$$

$$C_{24} = \frac{1}{(2\pi f_{0-10})^2 \times L_1} = \frac{10}{(2\pi \times 15.36 \times 10^3)^2} = 1.07 \times 10^{-9} F$$

Ta chọn: $C_{24} = 0.001\ \mu F$
Như vậy: $9.88\ KOhm < R_{24} < 14.02\ KOhm$
Ta chọn: R_{24}=10 KΩ

+ ***Băng 20.48KHz-40.96KHz:***

$$f_{0-11} = \frac{40.96 + 20.48}{2} = 30.72KHz$$

$$C_{25} = \frac{1}{(2\pi f_{0-11})^2 \times L_1} = \frac{10}{(2\pi \times 30.72 \times 10^3)^2} = 2.68 \times 10^{-10} F$$

Ta chọn: $C_{25} = 220\ pF$
Như vậy: $21.2\ KOhm < R_{25} < 30\ KOhm$
Ta chọn: R_{25}=22 KΩ

+ ***Băng 40.96KHz-81.92KHz:***

$$f_{0-12} = \frac{81.92 + 40.96}{2} = 61.44KHz$$

$$C_{26} = \frac{1}{(2\pi f_{0-12})^2 \times L_1} = \frac{10}{(2\pi \times 61.44 \times 10^3)^2} = 6.71 \times 10^{-11} F$$

Ta chọn: $C_{26} = 56\ pF$
Như vậy: $42.26\ KOhm < R_{26} < 59.76\ KOhm$
Ta chọn: R_{26}=47 KΩ

+ ***Băng 81,92KHz-163.84KHz:***

$$f_{0-13} = \frac{163.84 + 81.92}{2} = 122.88KHz$$

$$C_{27} = \frac{1}{(2\pi f_{0-13})^2 \times L_1} = \frac{10}{(2\pi \times 122.88 \times 10^3)^2} = 1.68 \times 10^{-11} F$$

Ta chọn: $C_{27} = 10\ pF$
Như vậy: $100\ KOhm < R_{27} < 141.42\ KOhm$
Ta chọn: R_{27}=120 KΩ

+ ***Băng 163.84KHz-327.68KHz:***

$$f_{0-14} = \frac{327.68 + 163.84}{2} = 245.76KHz$$

$$C_{28} = \frac{1}{(2\pi f_{0-14})^2 \times L_1} = \frac{10}{(2\pi \times 245.76 \times 10^3)^2} = 4.2 \times 10^{-12} F$$

Ta chọn: $C_{28} = 3\ pF$
Như vậy: $182.45\ KOhm < R_{28} < 258\ KOhm$
Ta chọn: R_{28}=200 KΩ

+ ***Băng 327.68KHz-655.36KHz:***

$$f_{0-15} = \frac{655.36 + 327.68}{2} = 491.52KHz$$

$$C_{29} = \frac{1}{(2\pi f_{0-15})^2 \times L_1} = \frac{10}{(2\pi \times 491.52 \times 10^3)^2} = 1.05 \times 10^{-12} F$$

Ta chọn: $C_{29} = 1\ pF$
Như vậy: $316\ KOhm < R_{30} < 447\ KOhm$
Ta chọn: R_{29}=330 KΩ

+ ***Băng 655.36KHz-1MHz:***

$$f_{0-16} = \frac{1000 + 655.36}{2} = 827.68KHz$$

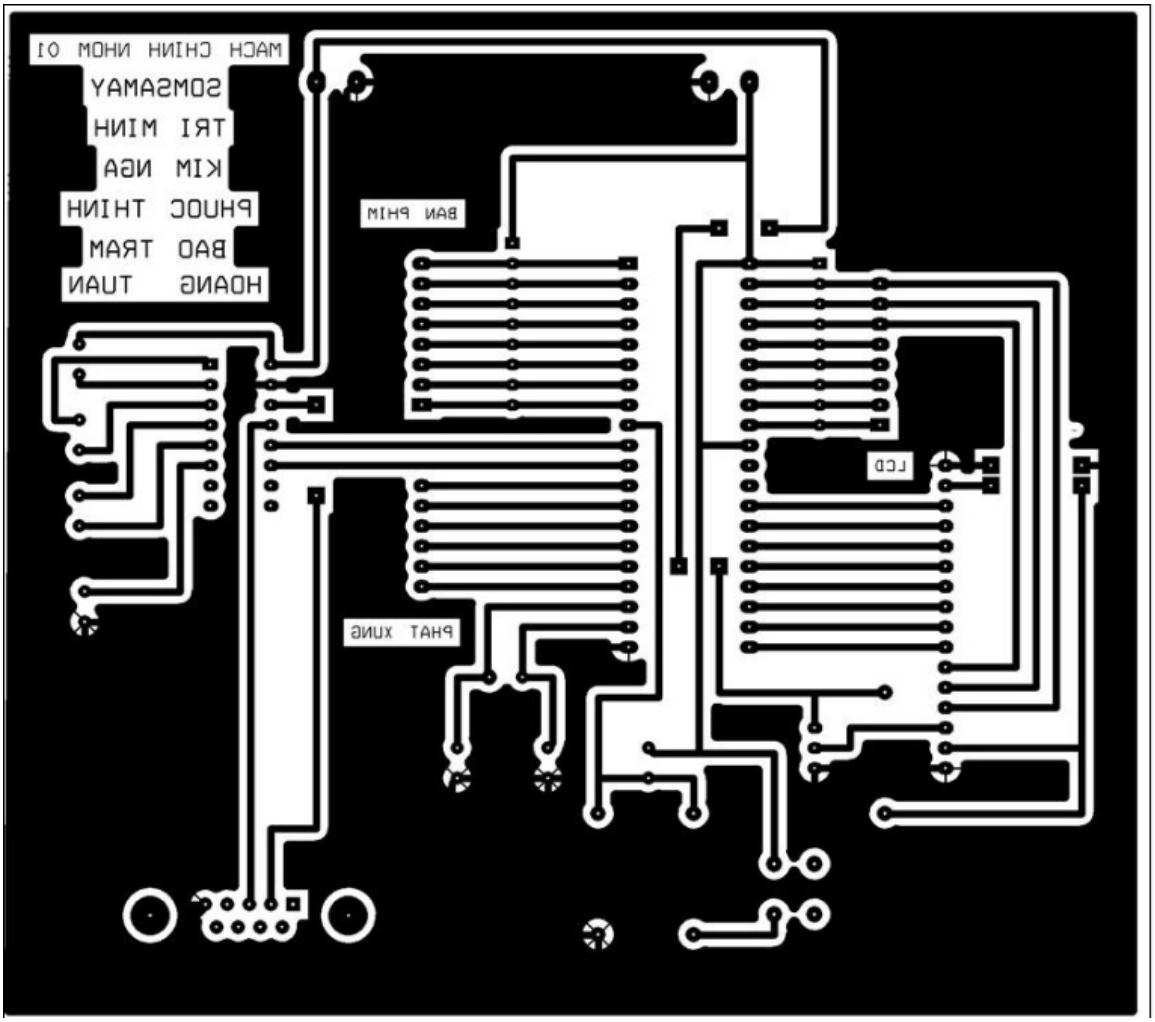
$$C_{30} = \frac{1}{(2\pi f_{0-16})^2 \times L_1} = \frac{10}{(2\pi \times 827.68 \times 10^3)^2} = 3.7 \times 10^{-13} F$$

Ta chọn: $C_{30} = 1\ pF$
Như vậy: $316\ KOhm < R_{30} < 447\ KOhm$
Ta chọn: R_{30}=330 KΩ.

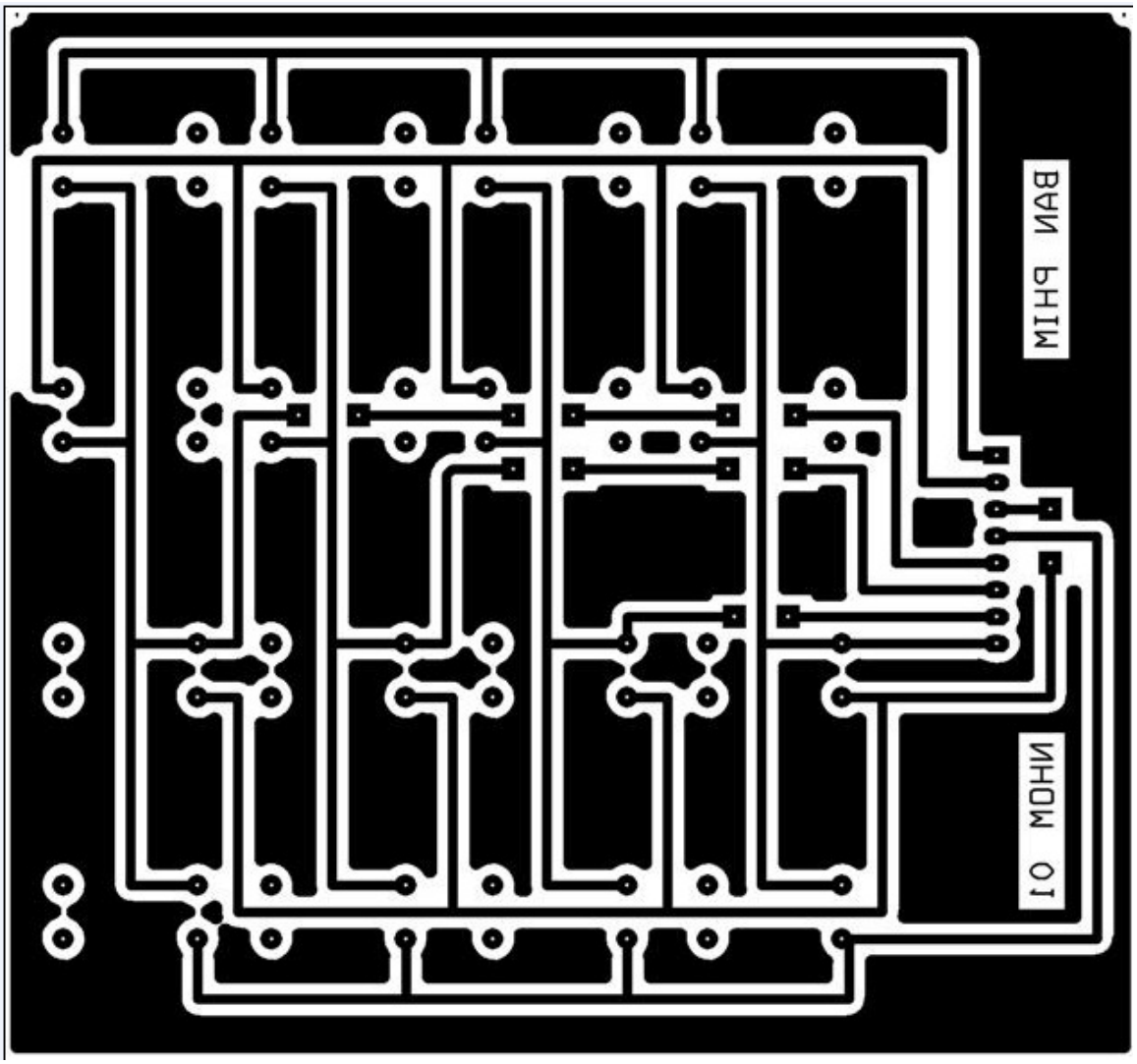
7. Chạy mô phỏng:

8. Vẽ mạch in:

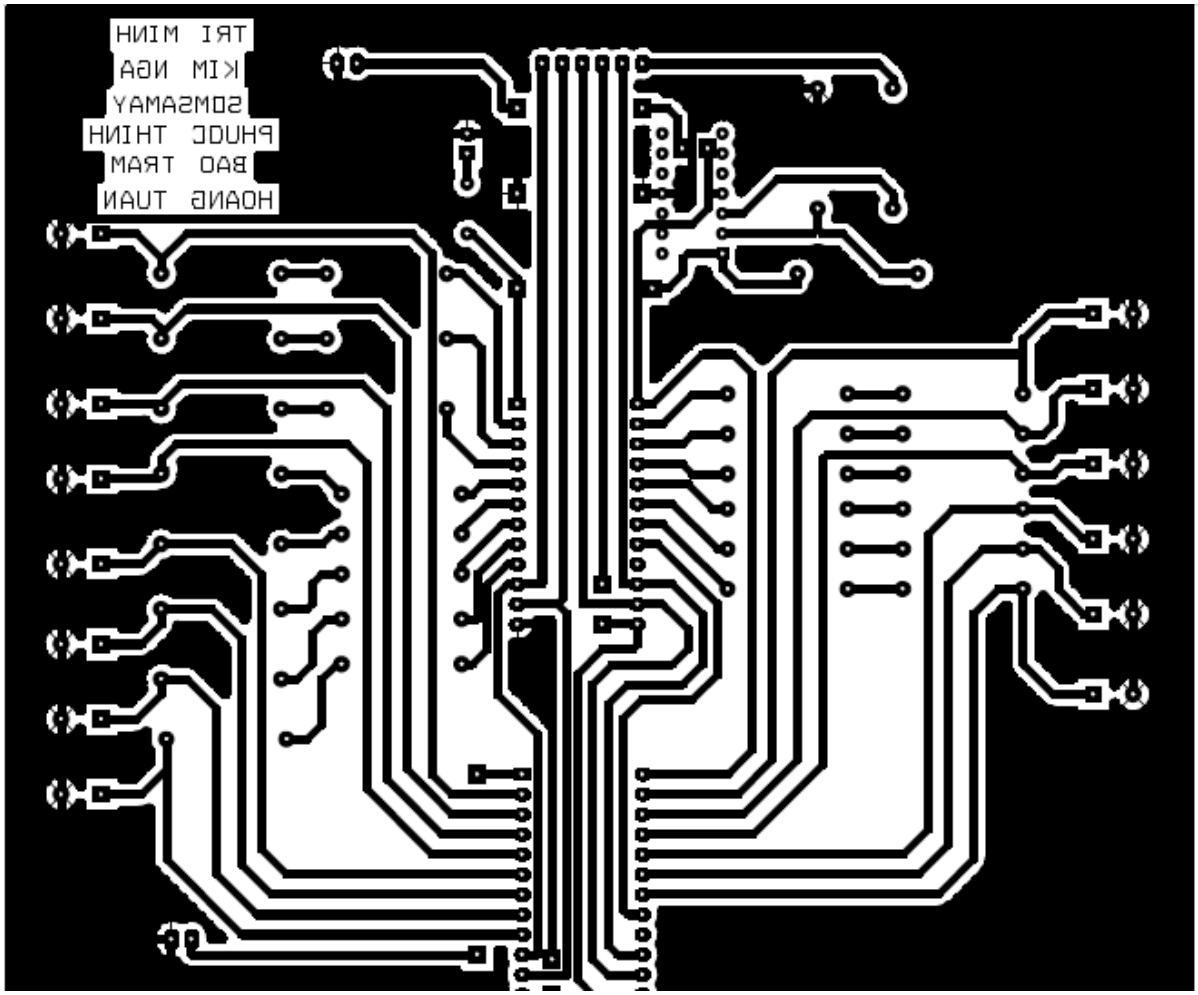
8.1 Khối xử lí trung tâm:



8.2 Khối bàn phím:



8.3 Khối phát dạng xung ra:



9. Kiểm tra linh kiện:

- Kiểm tra IC:
  - + Kiểm tra tương đối: IC nó chân bị cháy đen thường là những IC đã chết và cần thay mới.
  - + Kiểm tra chính xác: cấp nguồn, dùng chương trình test đơn giản để kiểm tra việc set các mức 0 và 1 của IC. Nếu IC hoạt động tốt thì ta chọn sử dụng IC này, nếu kết quả không đảm bảo mức điện áp TTL thì ta cần thay mới IC.
- Kiểm tra các tụ: dùng đồng hồ VOM để kiểm tra tụ hoạt động tốt hay xấu. Nếu tụ bị khô, bị rò hay bị đứt thì ta phải nhanh chóng thay thế bằng tụ mới tương đương để đảm bảo sự hoạt động của khối mạch chứa tụ điện đó, và cũng như đảm bảo cho hoạt động của toàn mạch.
- Kiểm tra trở: các trở thường ít bị hư hỏng trong quá trình sử dụng. Tuy nhiê n với những điện trở tải dòng lớn ta cần sử dụng các trở công suất để đảm bảo.
- Kiểm tra cuộn dây: kiểm tra sự chịu đựng dòng của cuộn dây có phù hợp với mạch thiết kế hay không, kiểm tra cuộn có bị đứt hay không bằng cách sử dụng VOM đặt ở thang đo trở. Nếu tình trạng của cuộn nằm ở một trong hai điều trên ta cần thay thế cuộn mới để mạch hoạt động tốt.
- Kiểm tra LCD:
  - + Kiểm tra tương đối: cấp nguồn và quan sát xem màu nền của LCD có sáng không. Nếu không thì LCD đã cháy và cần thay mới.

+ Kiểm tra tuyệt đối: sau khi cấp nguồn ta test thử LCD bằng chương trình hiển thị đơn giản. Nếu LCD hiển thị đúng thì chứng tỏ còn dùng được.

10. Lắp ráp mạch:

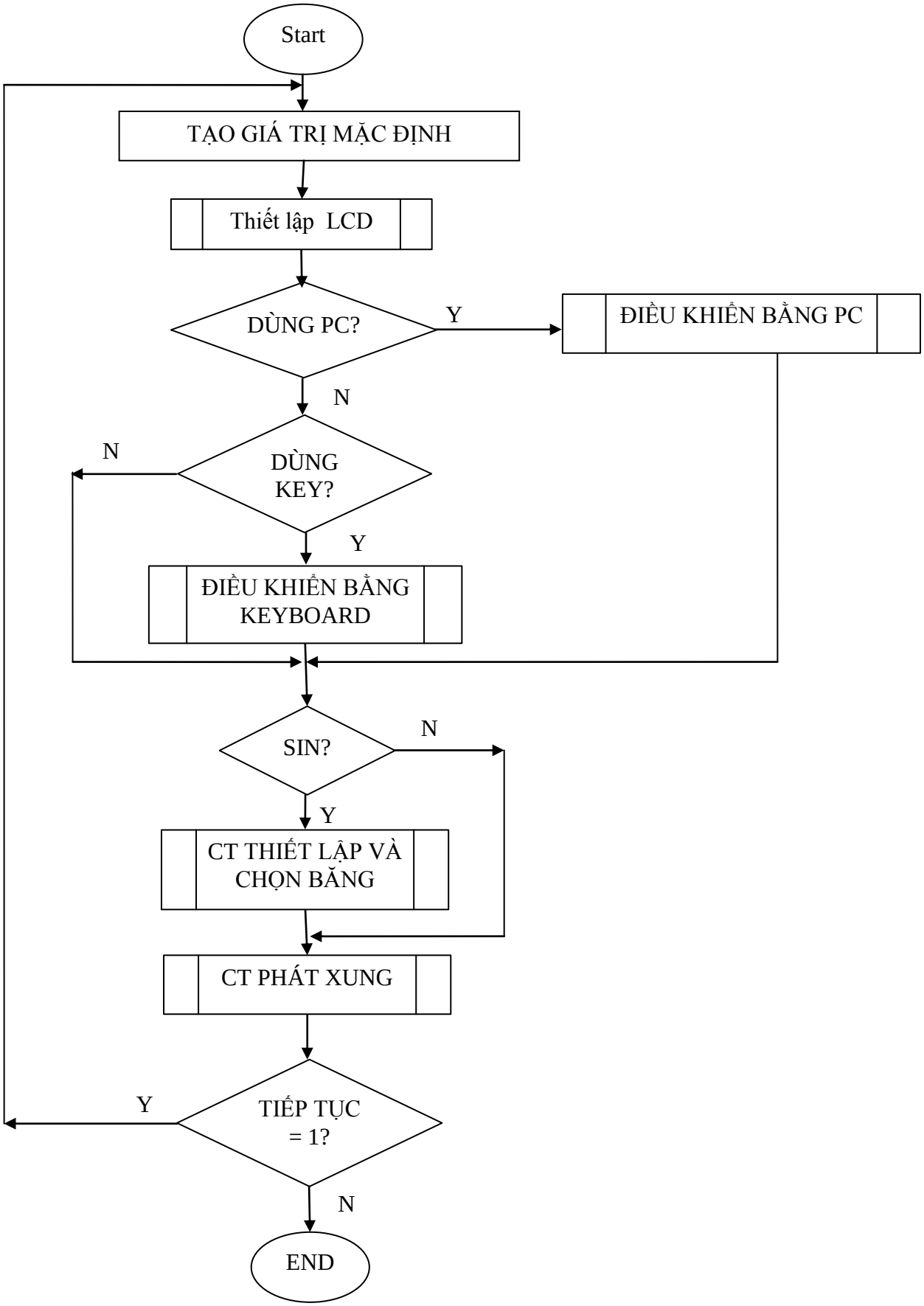
- Nối dây cấp nguồn cho khối mạch trung tâm gồm: vi điều khiển, max232.
- Nối dây cấp nguồn cho LCD.
- Nối dây cấp nguồn cho khối phát dạng song ngõ ra.
- Nối jump từ port 3 của vi điều khiển thuộc khối mạch trung tâm vào đầu vào của khối mạch phát dạng sóng ngõ ra.
- Nối dây cổng COM với máy tính để thực hiện giao tiếp.
- Nối jump từ port 1 của vi điều khiển thuộc khối mạch trung tâm vào khối bàn phím để thực hiện việc giao tiếp.

11. Kiểm tra mạch:

- Quan sát toàn bộ một lượt mạch in, kiểm tra xem các đường đồng có còn tốt hay không, có đảm bảo kích thước dẫn dòng hay không để xử lý kịp thời.
- Kiểm tra khối LCD có hoạt động tốt không: Nếu không hiển thị gì thì ta kiểm tra xem port đó đã được kéo dòng hay chưa. Nếu chưa thì cần phải kéo trở nâng dòng do LCD hút dòng mạnh, nếu đã kéo trở thì ta tinh chỉnh lại biến trở constrast xem có cải thiện được không. Nếu hai biện pháp trên đã dùng mà không cải thiện thì lỗi nằm ở phần chương trình.
- Kiểm tra khối bàn phím: Nếu bấm phím không nhập được giá trị thì phải kiểm tra xem có switch nào bị hỏng không, đã thực hiện kéo trở nâng dòng cho khối bàn phím chưa. Nếu cả hai đều không có thì lỗi nằm ở phần chương trình.
- Kiểm tra mạch phát xung: Nếu xung ra phát không ổn định và không đúng với yêu cầu thì ta cần kiểm tra lại khối thạch anh xem có còn dùng được hay không. Nếu dạng sóng sin phát ra bị méo thì cần kiểm tra xem việc thực hiện chuyển mạch đã đúng bộ lọc hay chưa và trở với tụ cho bộ lọc đó đã chọn đúng hay chưa. Nếu sau khi kiểm tra tất cả đều đạt yêu cầu nhưng xung phát ra vẫn không đúng yêu cầu thì chứng tỏ lỗi nằm ở phần chương trình.
- Kiểm tra giao tiếp máy tính: Nếu việc giao tiếp không thực hiện được với thạch anh 24MHz thì phải kiểm tra xem vi điều khiển đang dùng có đúng là P89V51RD2 hay không, bên cạnh đó cần kiểm tra lại dây cổng COM có còn dùng được hay không, chiều dài dây đã hợp lý hay chưa. Nếu đã thực hiện hết các công việc trên mà việc giao tiếp máy tính vẫn không cải thiện được thì chứng tỏ lỗi nằm ở phần chương trình.

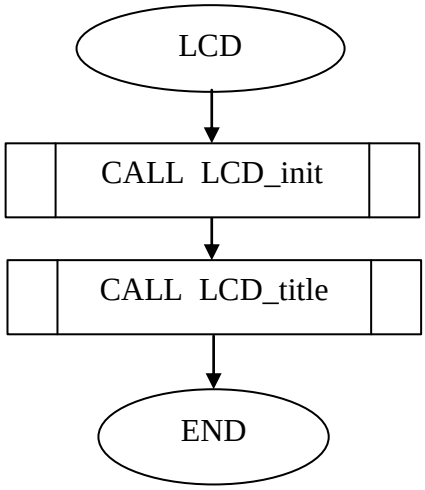
12. Viết lưu đồ thuật toán:

12. 1 Lưu đồ thuật toán chương trình chính:

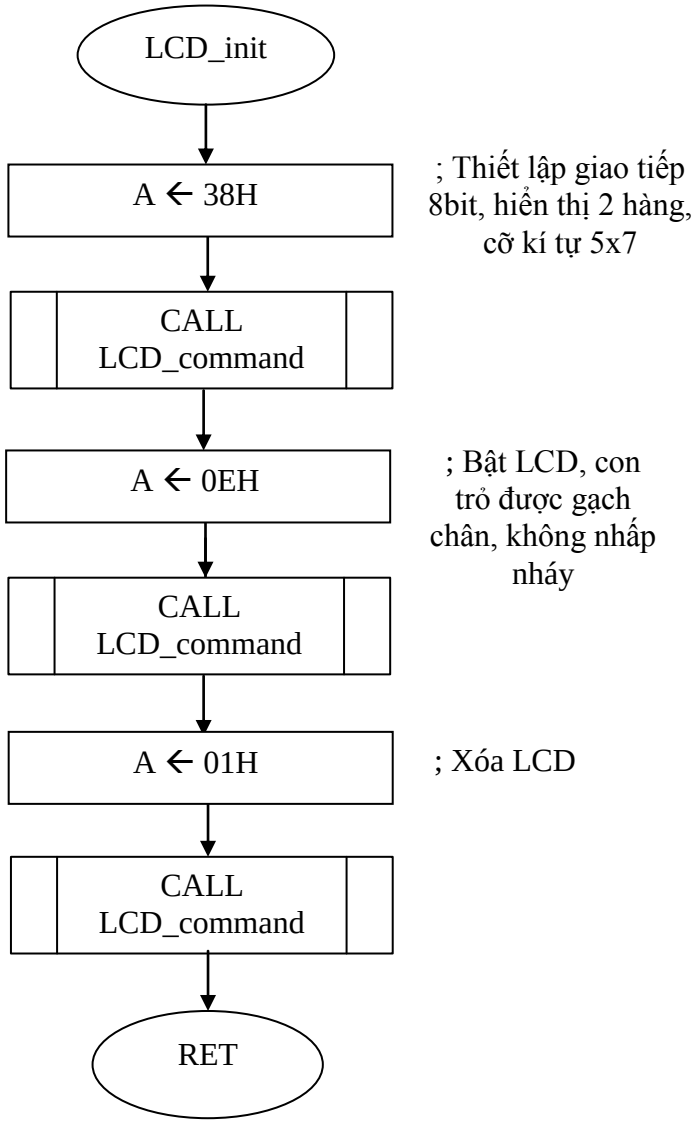

Start
TẠO GIÁ TRỊ MẶC ĐỊNH
Thiết lập LCD
DÙNG PC?
Y
ĐIỀU KHIỂN BẰNG PC
N
DÙNG KEY?
N
Y
ĐIỀU KHIỂN BẰNG KEYBOARD
SIN?
N
Y
CT THIẾT LẬP VÀ CHỌN BĂNG
CT PHÁT XUNG
TIẾP TỤC = 1?
Y
N
END

12.2 Lưu đồ thuật toán chương trình con thiết lập LCD:

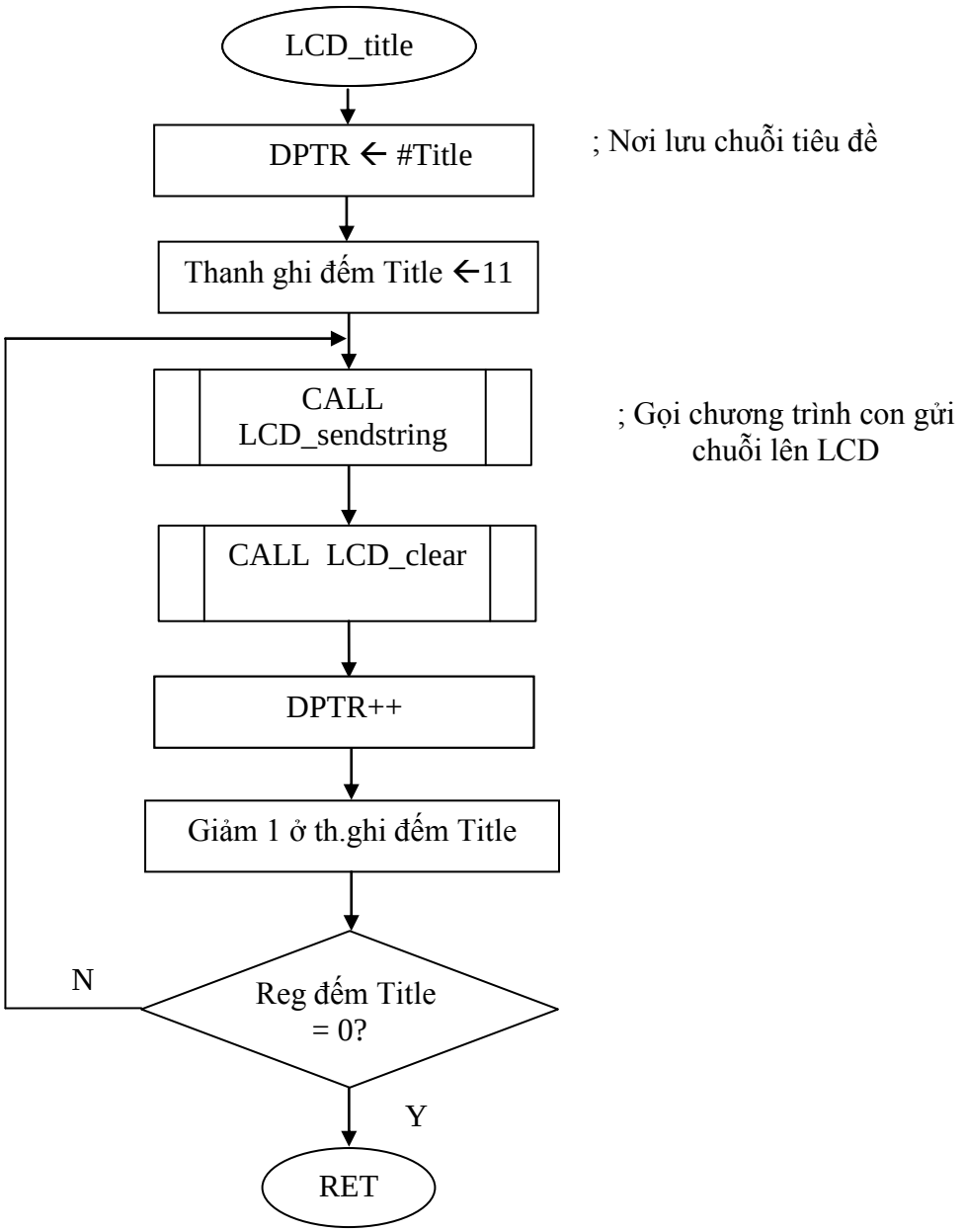
SƠ ĐỒ KHỐI
Chương trình con LCD



SƠ ĐỒ KHỐI
Chương trình con KHỞI TAO LCD

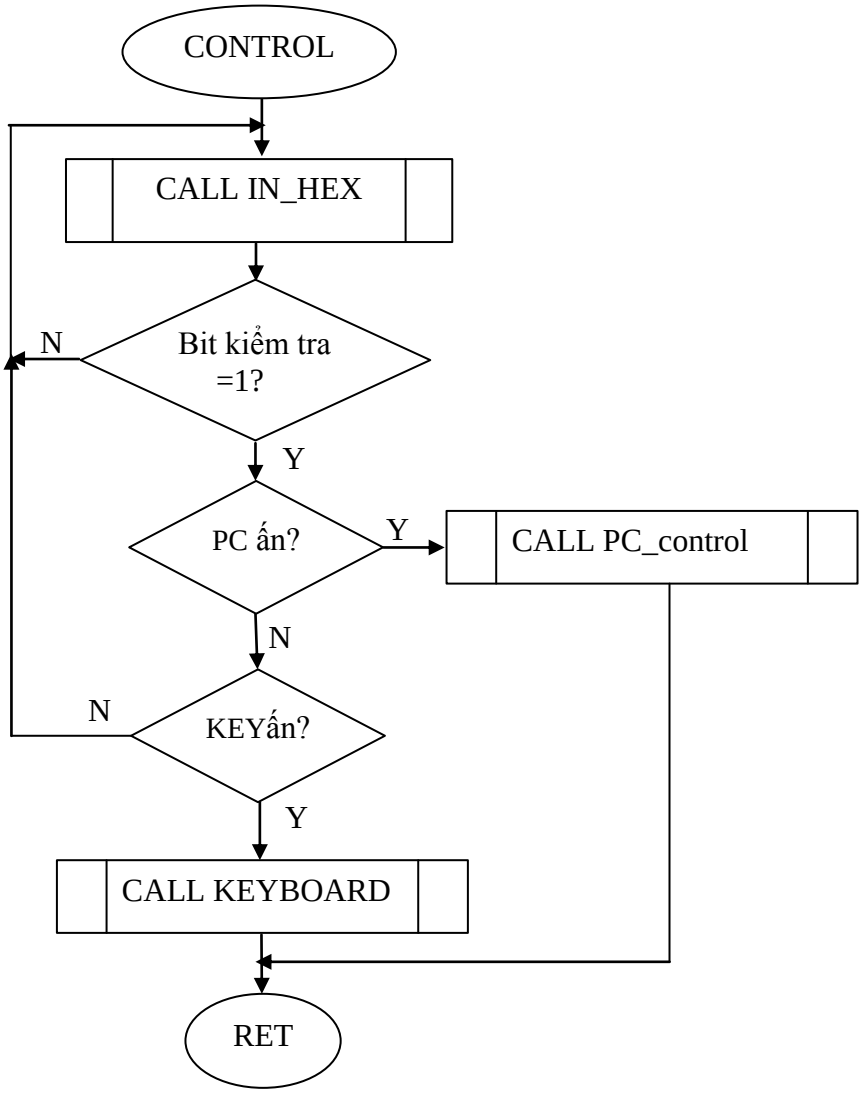


SƠ ĐỒ KHỐI
Chương trình con TIÊU ĐỀ LCD



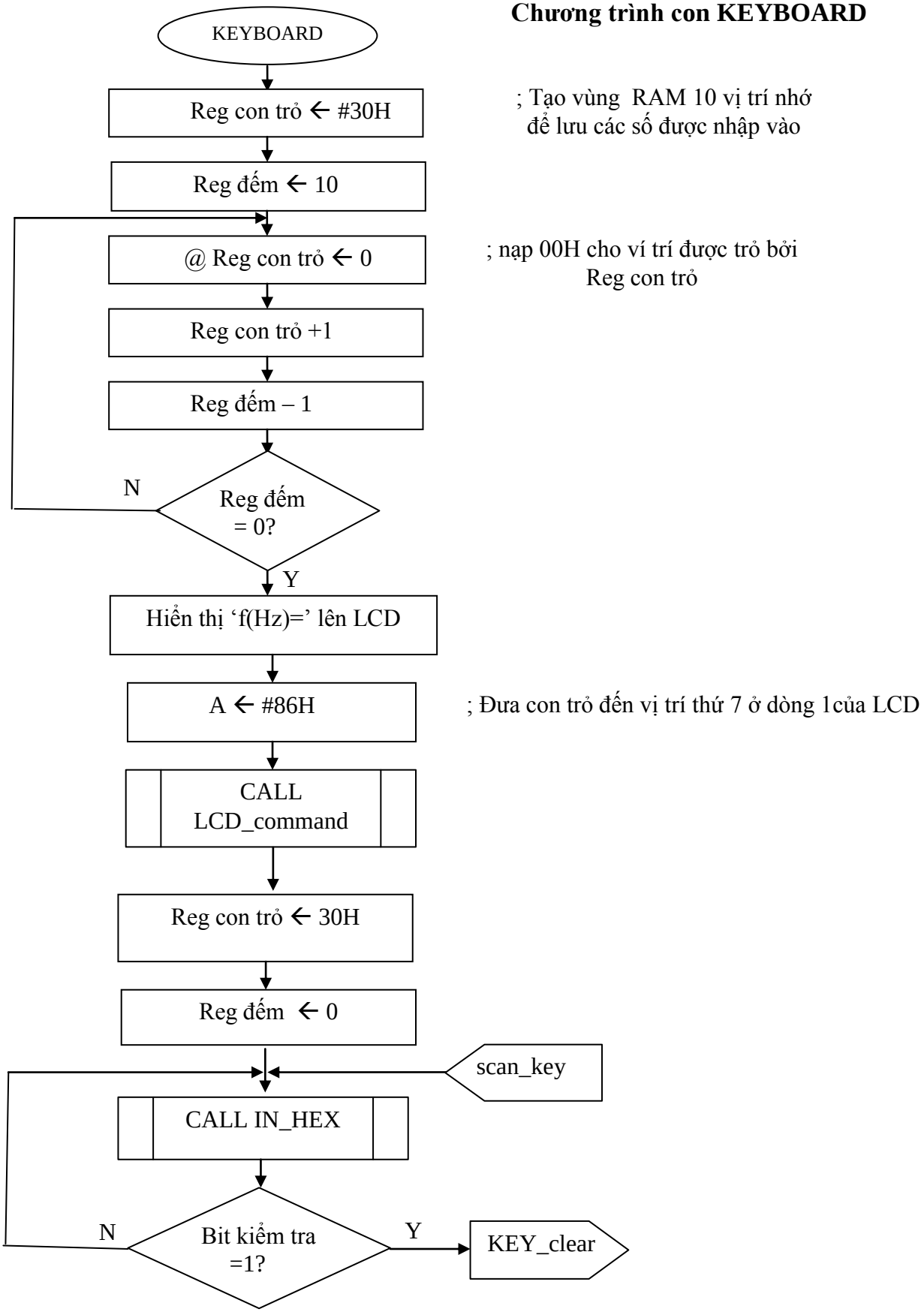
12.3 Lưu đồ thuật toán chương trình con điều khiển CONTROL: (Bảo Trâm)

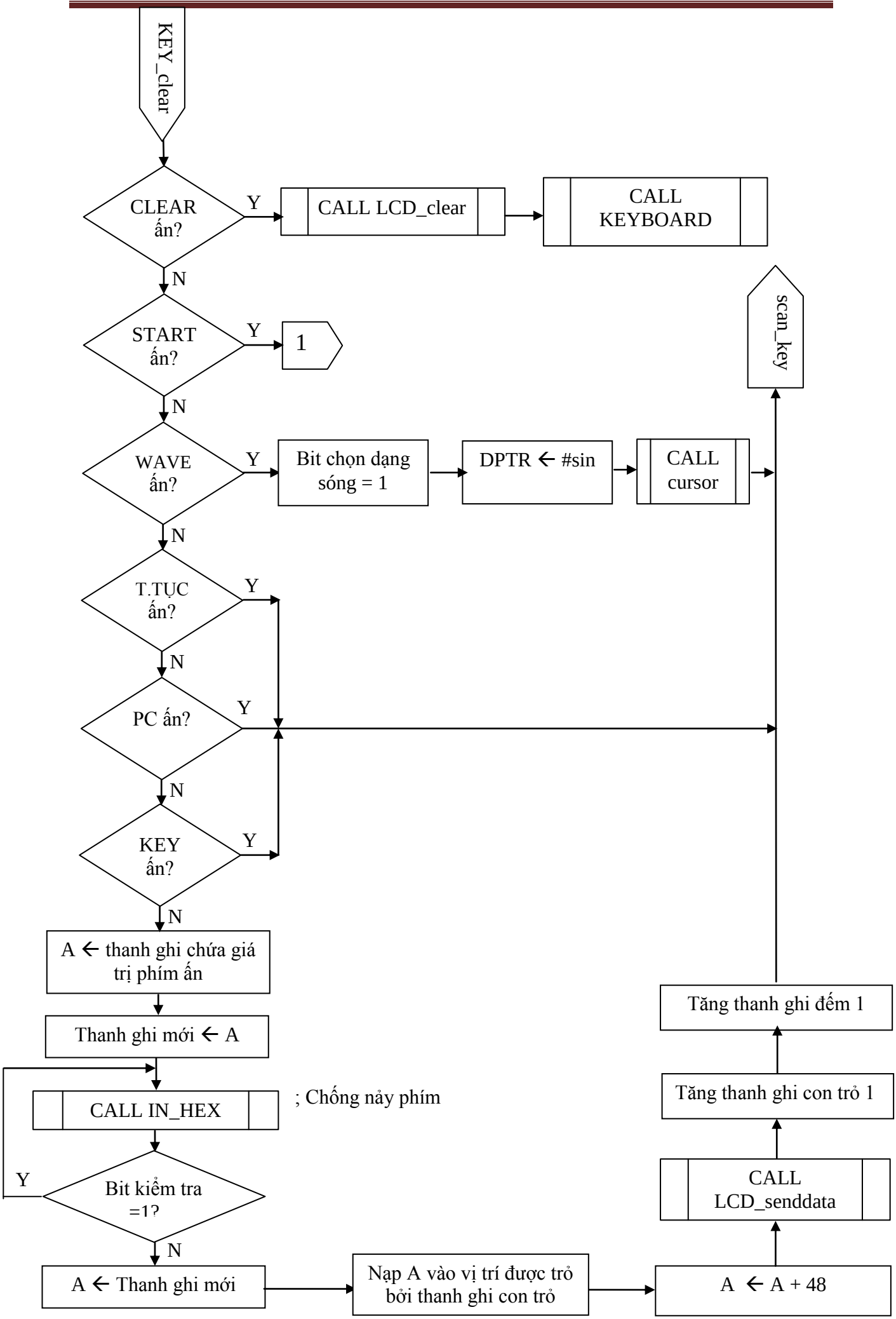
- Chương trình điều khiển dùng PC_control hay KEYBOARD

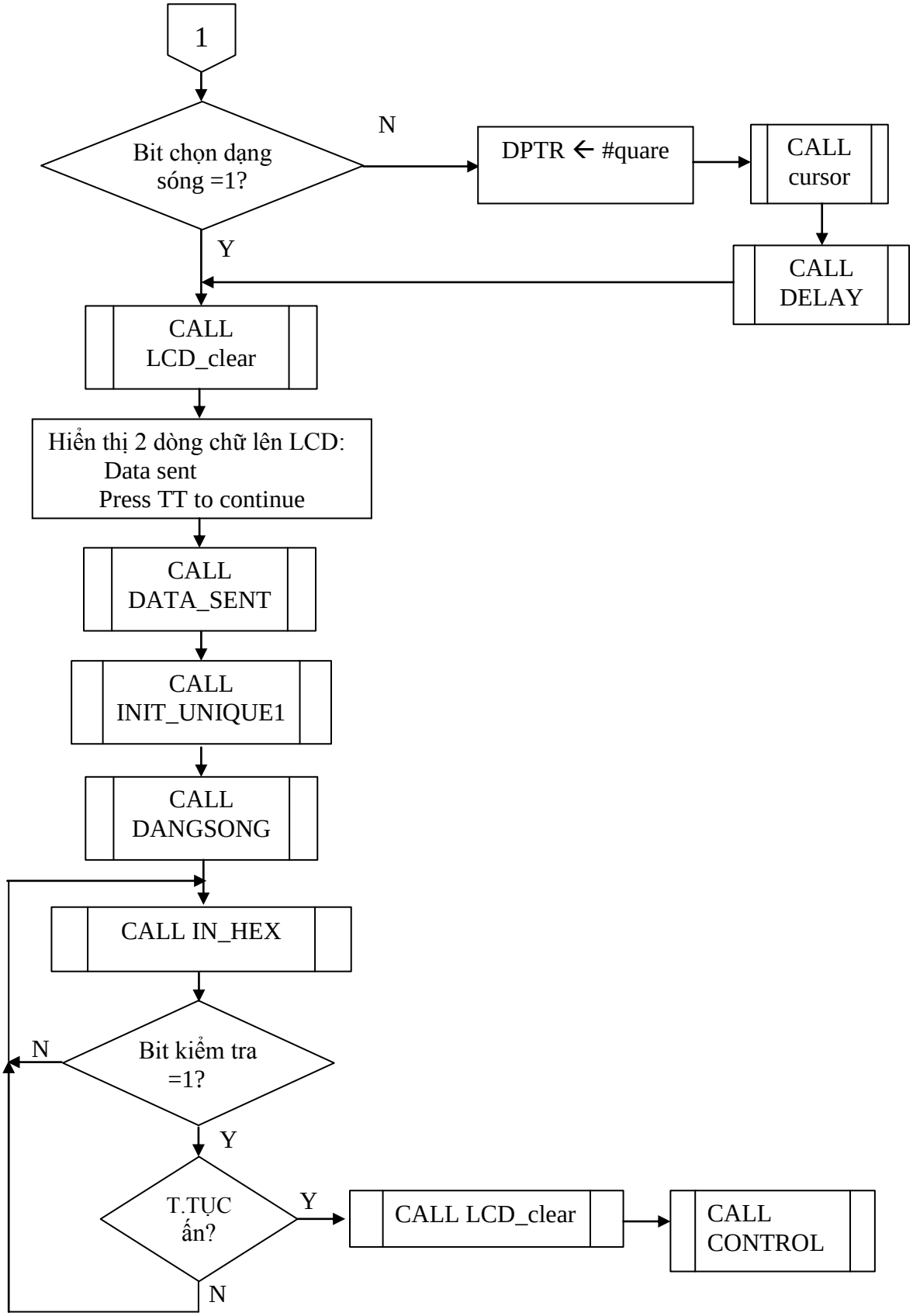


12.4 Lưu đồ thuật toán chương trình con KEYBOARD: (Bảo Trâm)

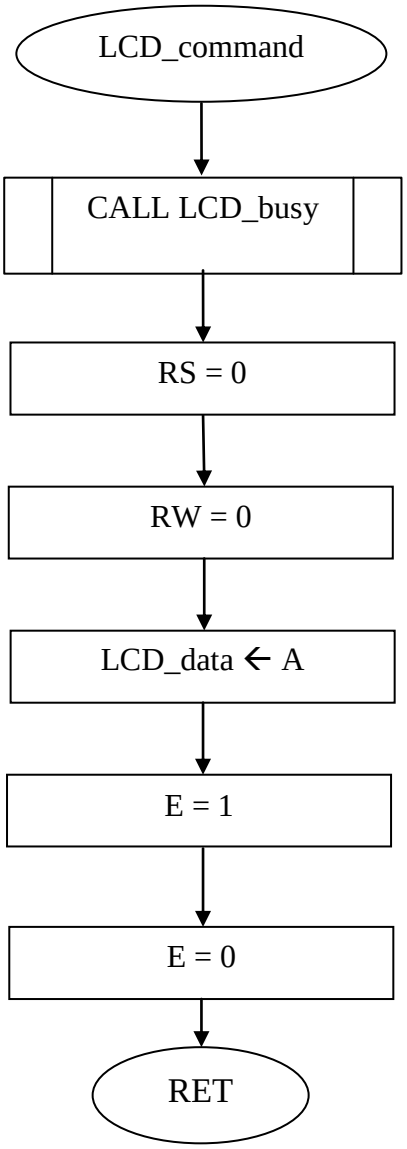
SƠ ĐỒ KHỐI
Chương trình con KEYBOARD



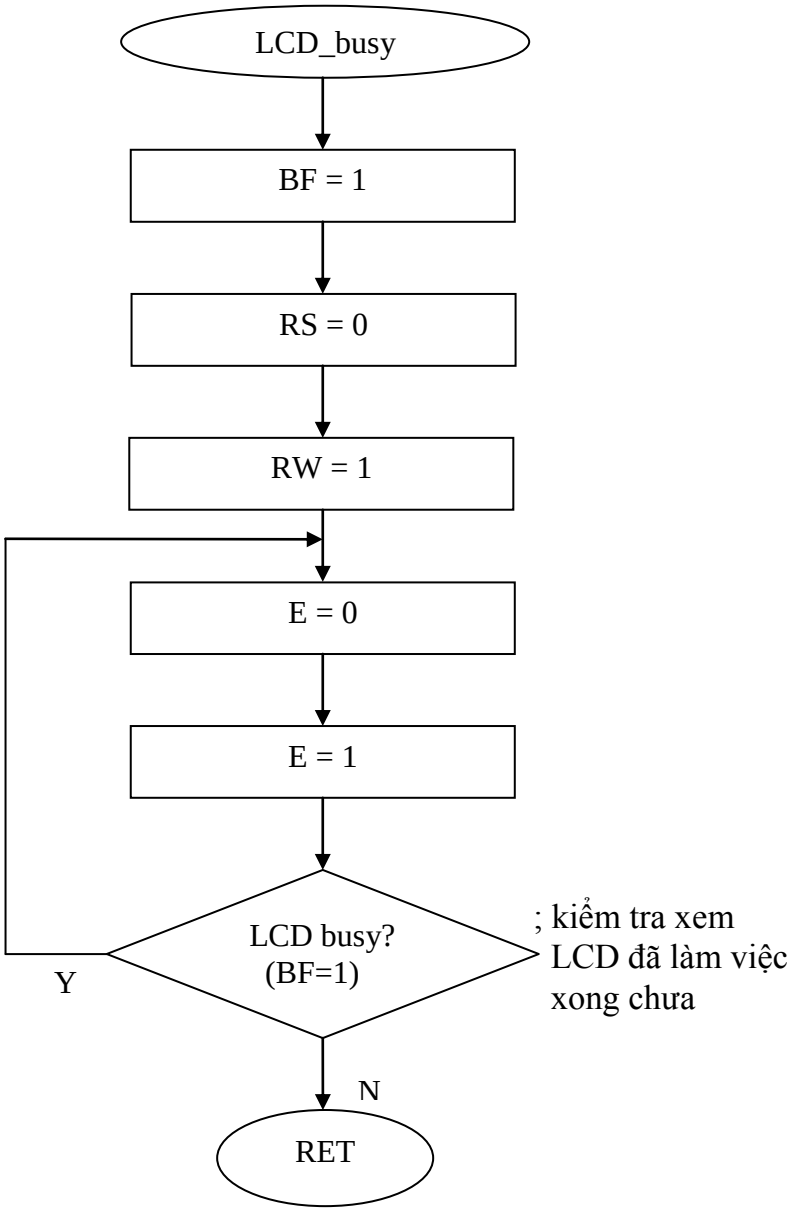

KEY_clear
CLEAR ấn?
Y
CALL LCD_clear
CALL KEYBOARD
N
START ấn?
Y
1
N
WAVE ấn?
Y
Bit chọn dạng sóng = 1
DPTR ← #sin
CALL cursor
N
T.TỤC ấn?
Y
N
PC ấn?
Y
N
KEY ấn?
Y
N
A ← thanh ghi chứa giá trị phím ấn
Thanh ghi mới ← A
CALL IN_HEX
; Chống nảy phím
Y
Bit kiểm tra =1?
N
A ← Thanh ghi mới
Nạp A vào vị trí được trỏ bởi thanh ghi con trỏ
A ← A + 48
CALL LCD_senddata
Tăng thanh ghi con trỏ 1
Tăng thanh ghi đếm 1
scan_key


1
Bit chọn dạng sóng =1?
N
DPTR ← #quare
CALL cursor
CALL DELAY
Y
CALL LCD_clear
Hiển thị 2 dòng chữ lên LCD:
Data sent
Press TT to continue
CALL DATA_SENT
CALL INIT_UNIQUE1
CALL DANGSONG
CALL IN_HEX
Bit kiểm tra =1?
N
Y
T.TỤC ấn?
Y
CALL LCD_clear
CALL CONTROL
N

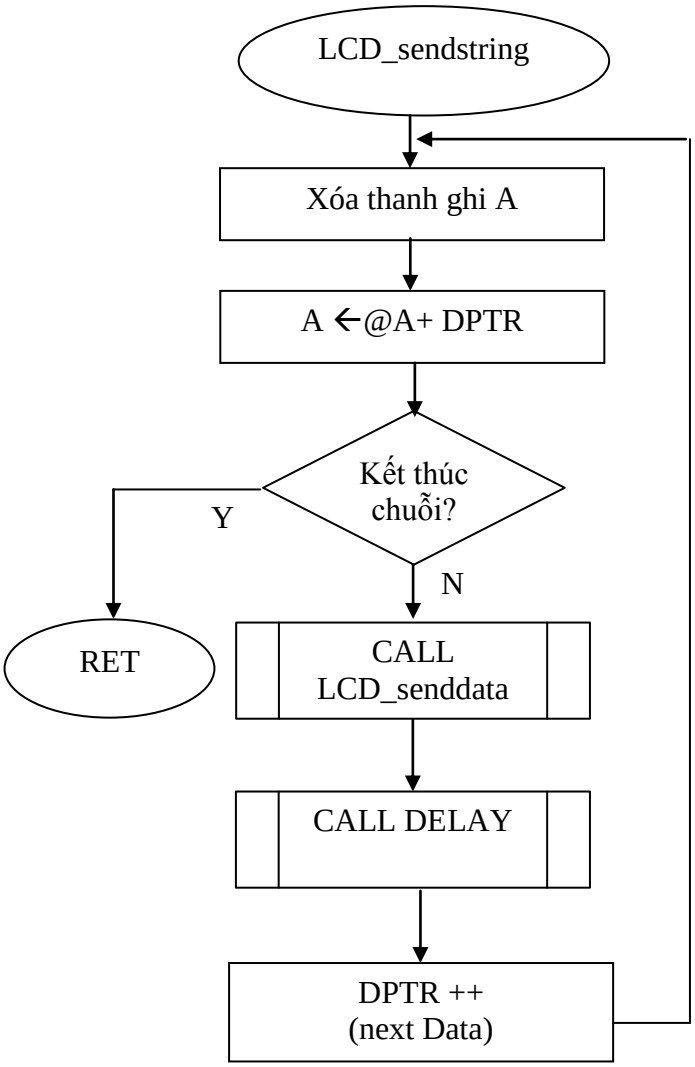
SƠ ĐỒ KHỐI
Chương trình con LCD_command
(gởi lệnh lên LCD)



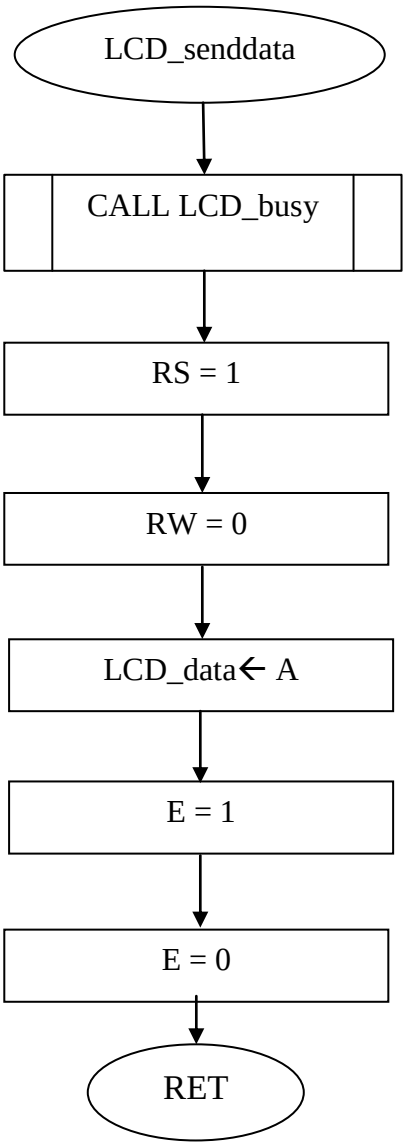
SƠ ĐỒ KHỐI
Chương trình con LCD_busy
(Kiểm tra cờ Busy – bit thứ 7 của data)



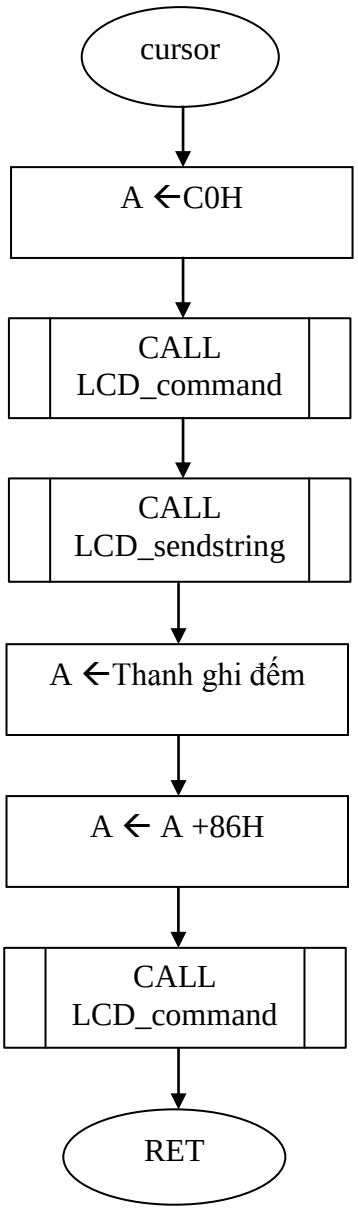
SƠ ĐỒ KHỐI
Chương trình con LCD_sendstring
(gởi chuỗi dữ liệu lên LCD)



SƠ ĐỒ KHỐI
Chương trình con LCD_senddata
(gởi dữ liệu lên LCD)



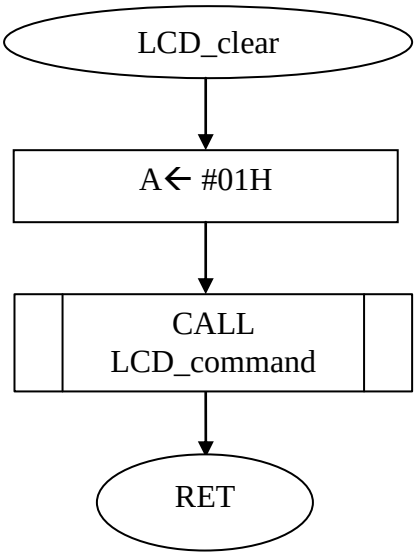
SƠ ĐỒ KHỐI
Chương trình con cursor
(dịch chuyển vị trí con trỏ)



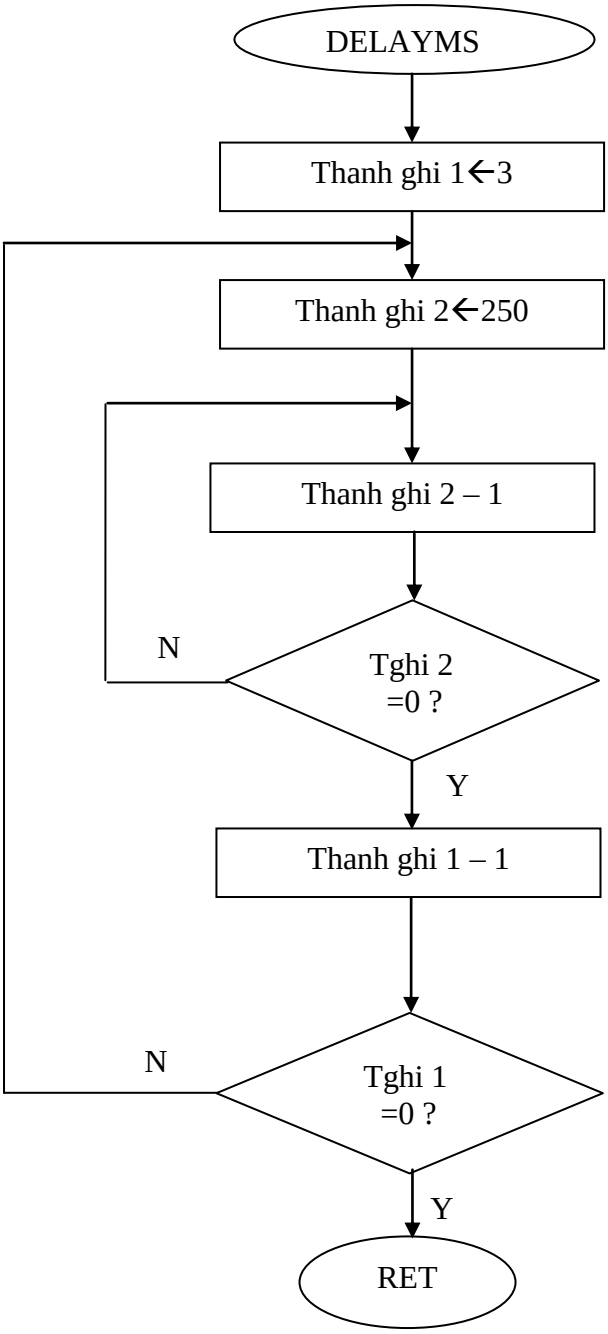
; Đưa con trỏ đến vị trí đầu tiên của dòng thứ 2 trên LCD

; Đưa con trỏ đến vị trí ngay sau chữ số cuối cùng trong dãy đã nhập

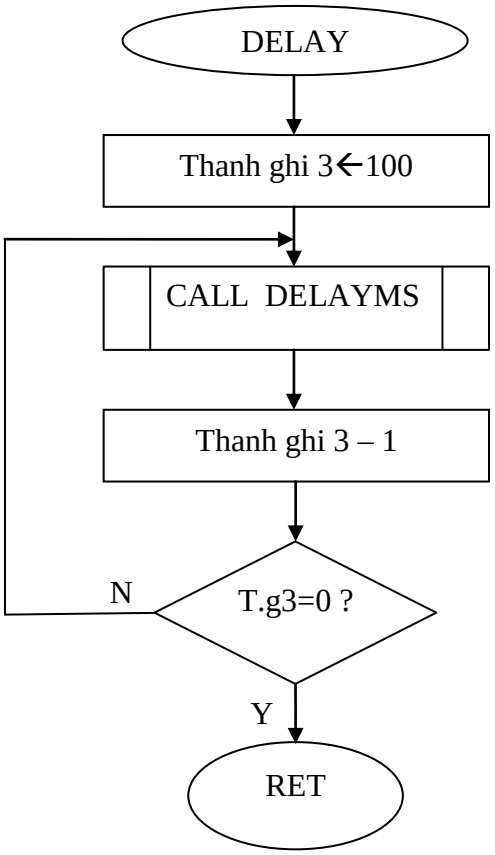
SƠ ĐỒ KHỐI
Chương trình con LCD_clear
(xóa màn hình LCD)



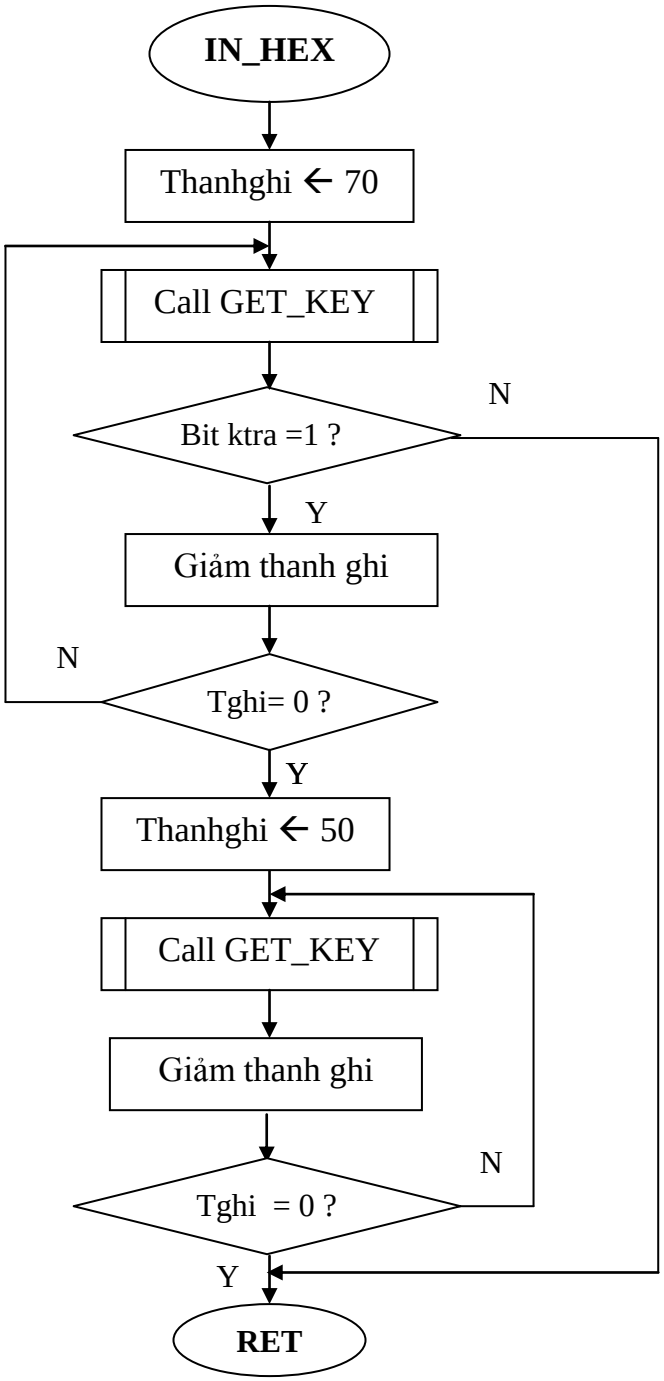
SƠ ĐỒ KHỐI
Chương trình con DELAYMS
(Tạo thời gian trễ 1ms)

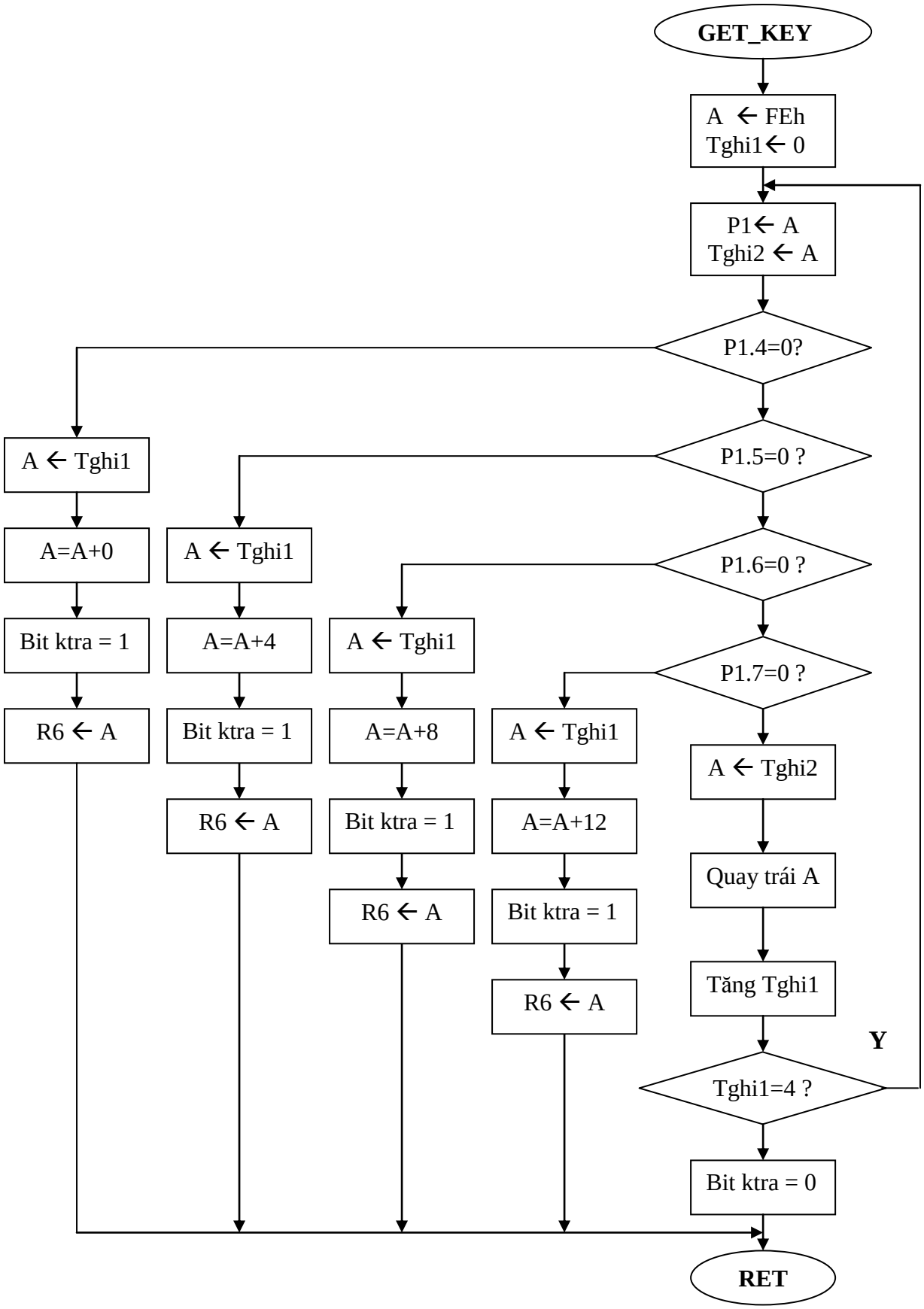


SƠ ĐỒ KHỐI
Chương trình con DELAY
(Tạo thời gian trễ 100ms)

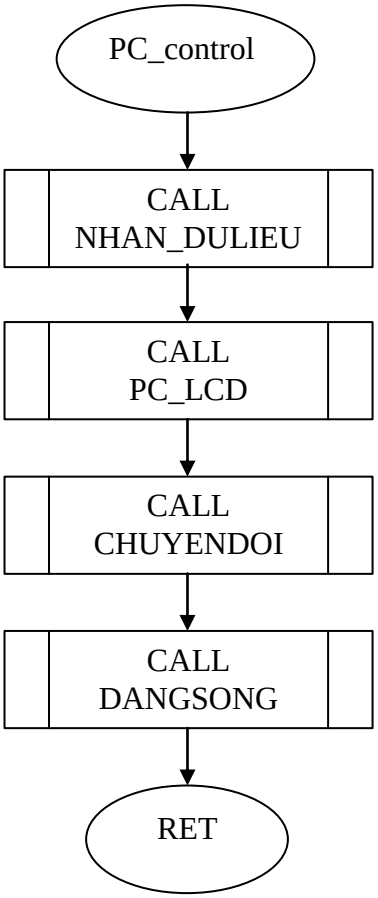


Sơ đồ khối chương trình con quét phím: (Phước Thịnh)

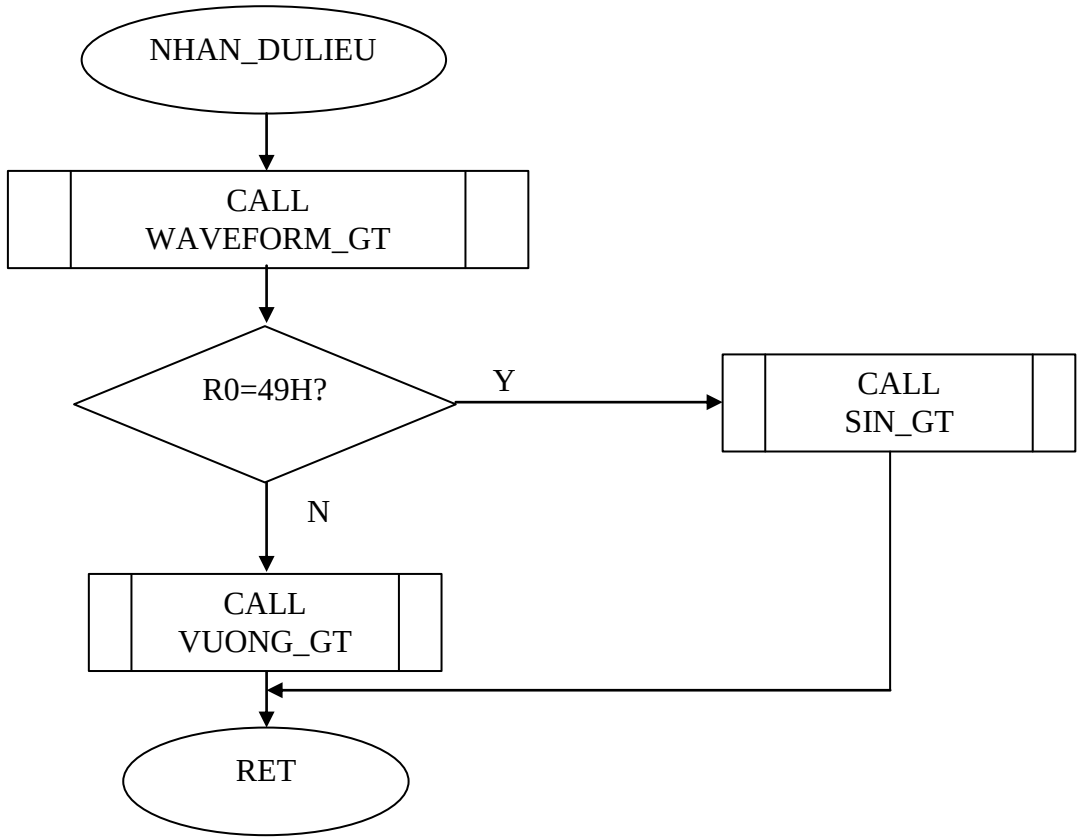



GET_KEY
A ← FEh
Tghi1← 0
P1← A
Tghi2 ← A
P1.4=0?
A ← Tghi1
P1.5=0 ?
A=A+0
A ← Tghi1
P1.6=0 ?
Bit ktra = 1
A=A+4
A ← Tghi1
P1.7=0 ?
R6 ← A
Bit ktra = 1
A=A+8
A ← Tghi1
A ← Tghi2
R6 ← A
Bit ktra = 1
A=A+12
Quay trái A
R6 ← A
Bit ktra = 1
Tăng Tghi1
R6 ← A
Y
Tghi1=4 ?
Bit ktra = 0
RET

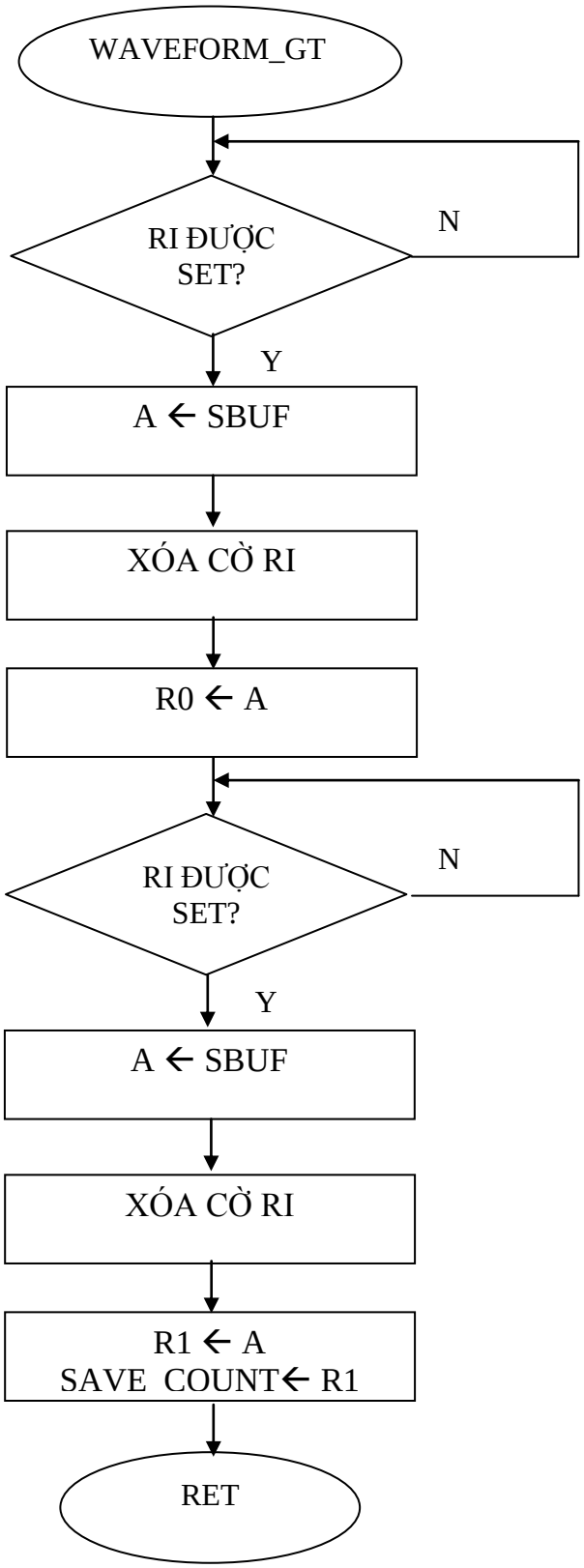
12.5 Lưu đồ thuật toán chương trình con PC_control: (Trí Minh)



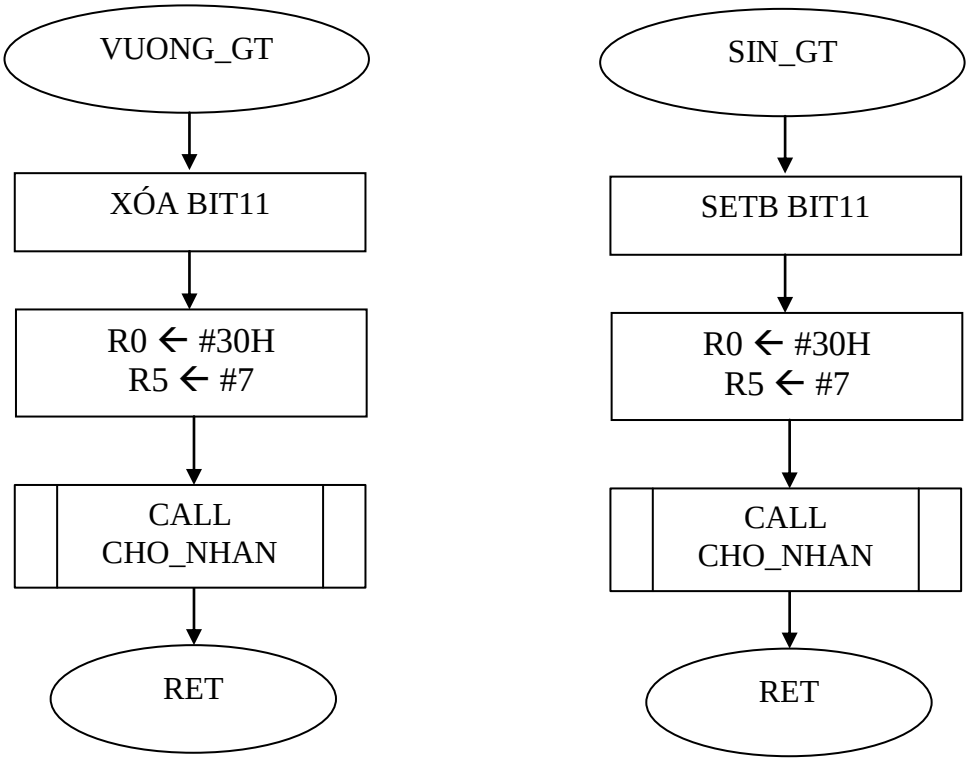
+ Chương trình con NHANDULIEU:



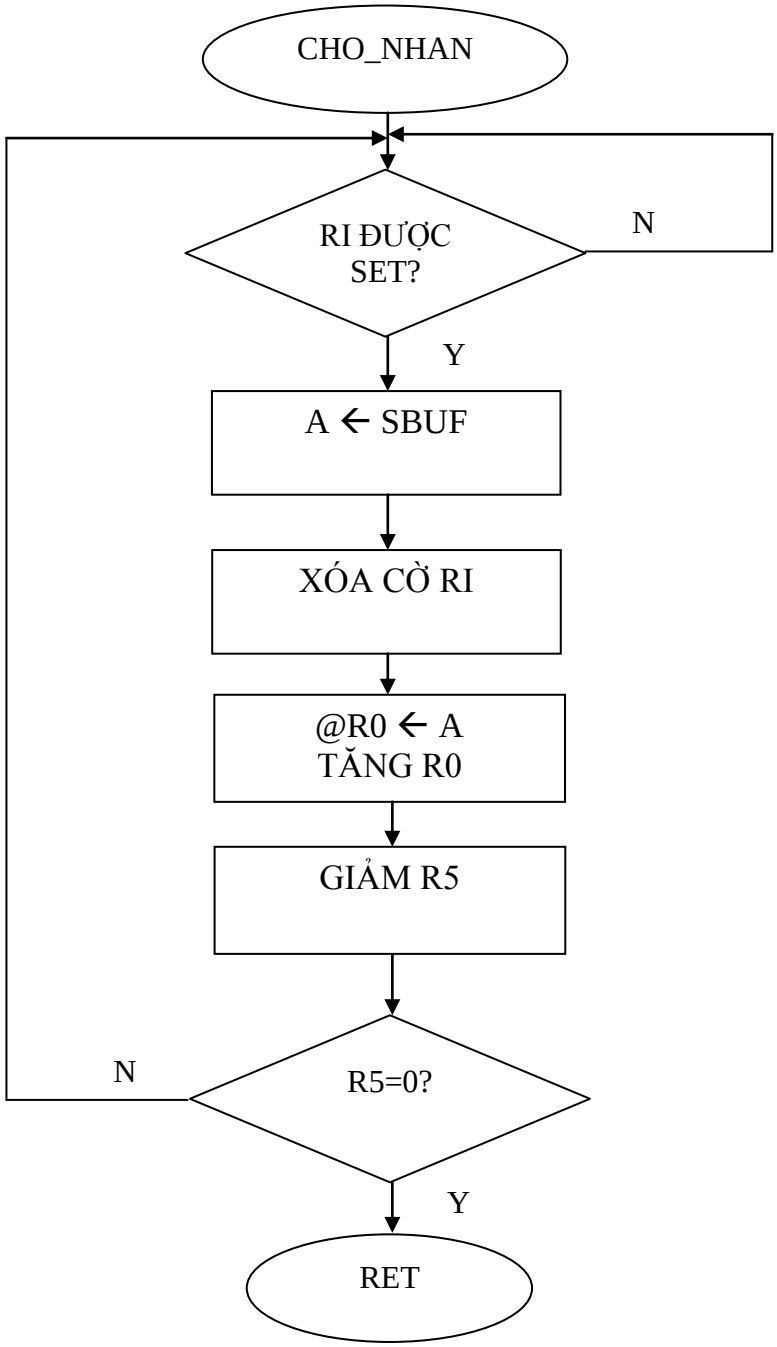
+ Chương trình con WAVEFORM_GT:



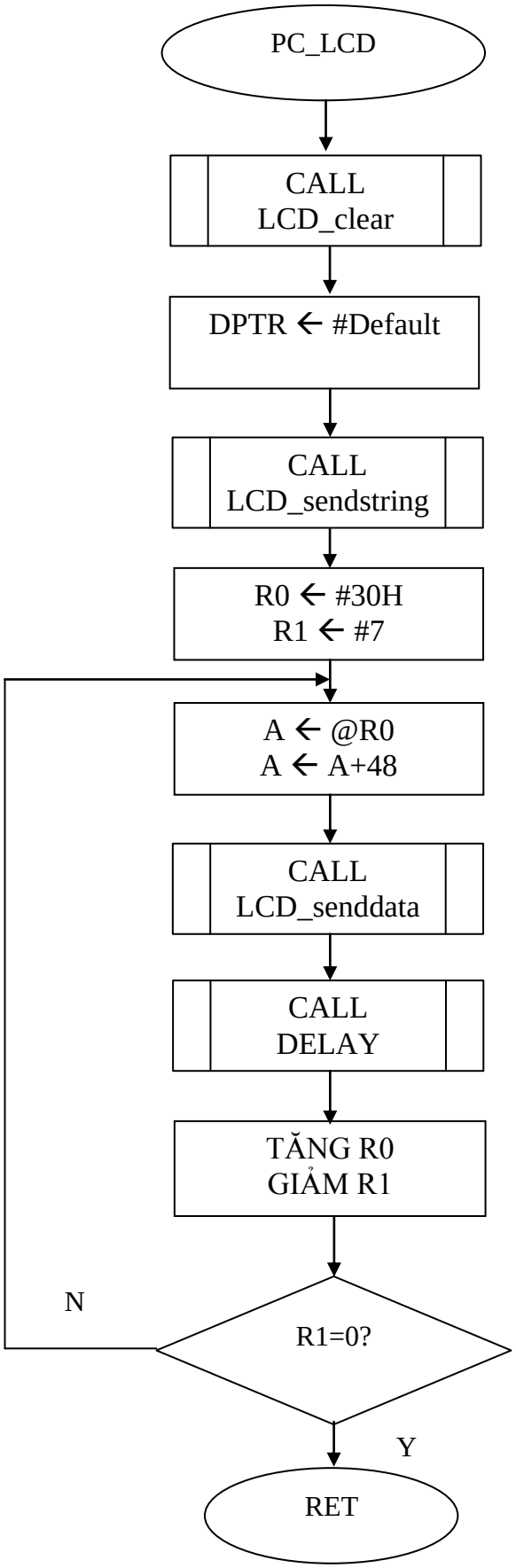
+ Chương trình con VUONG_GT hoặc SIN_GT:



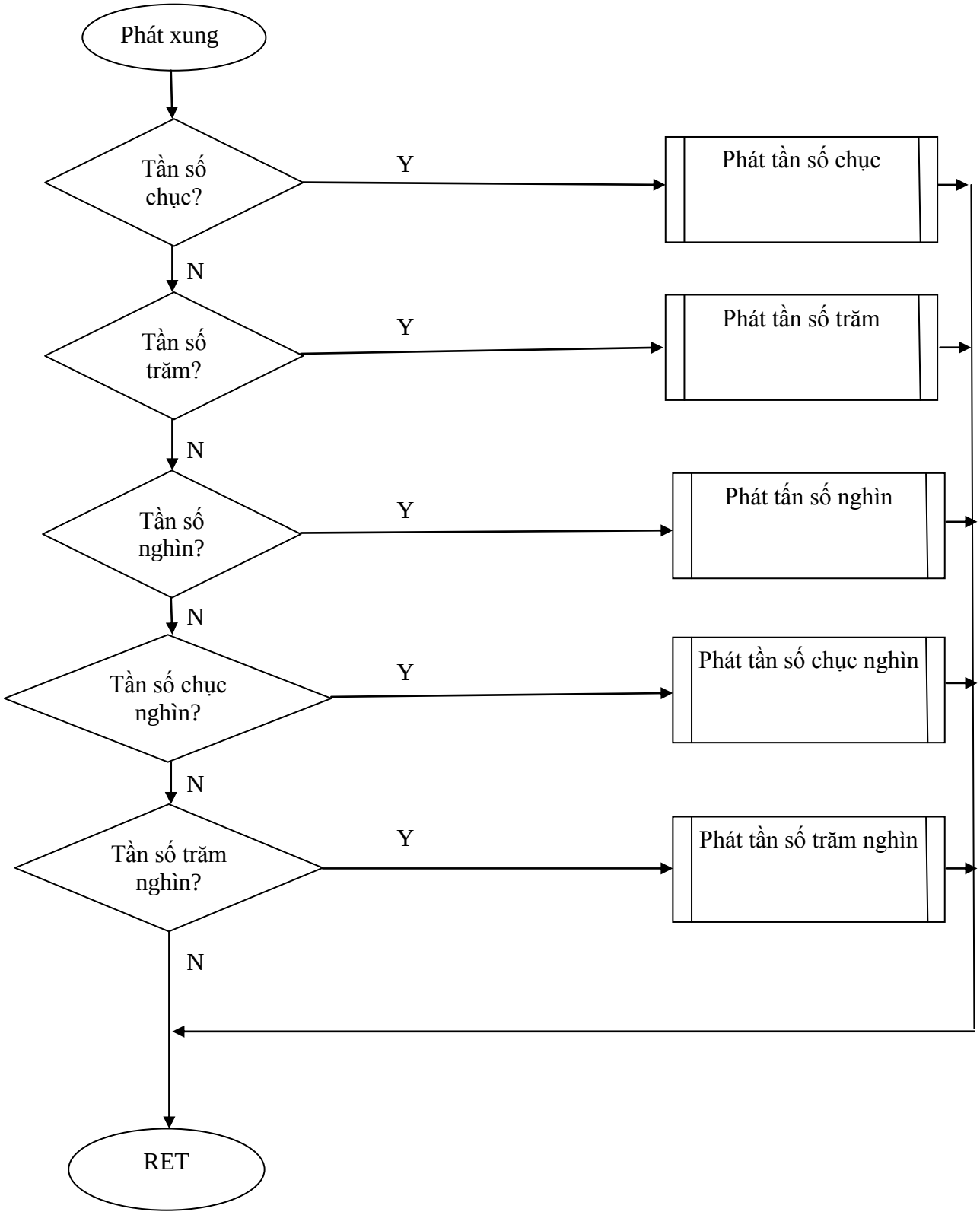
+ Chương trình con CHO_NHAN:



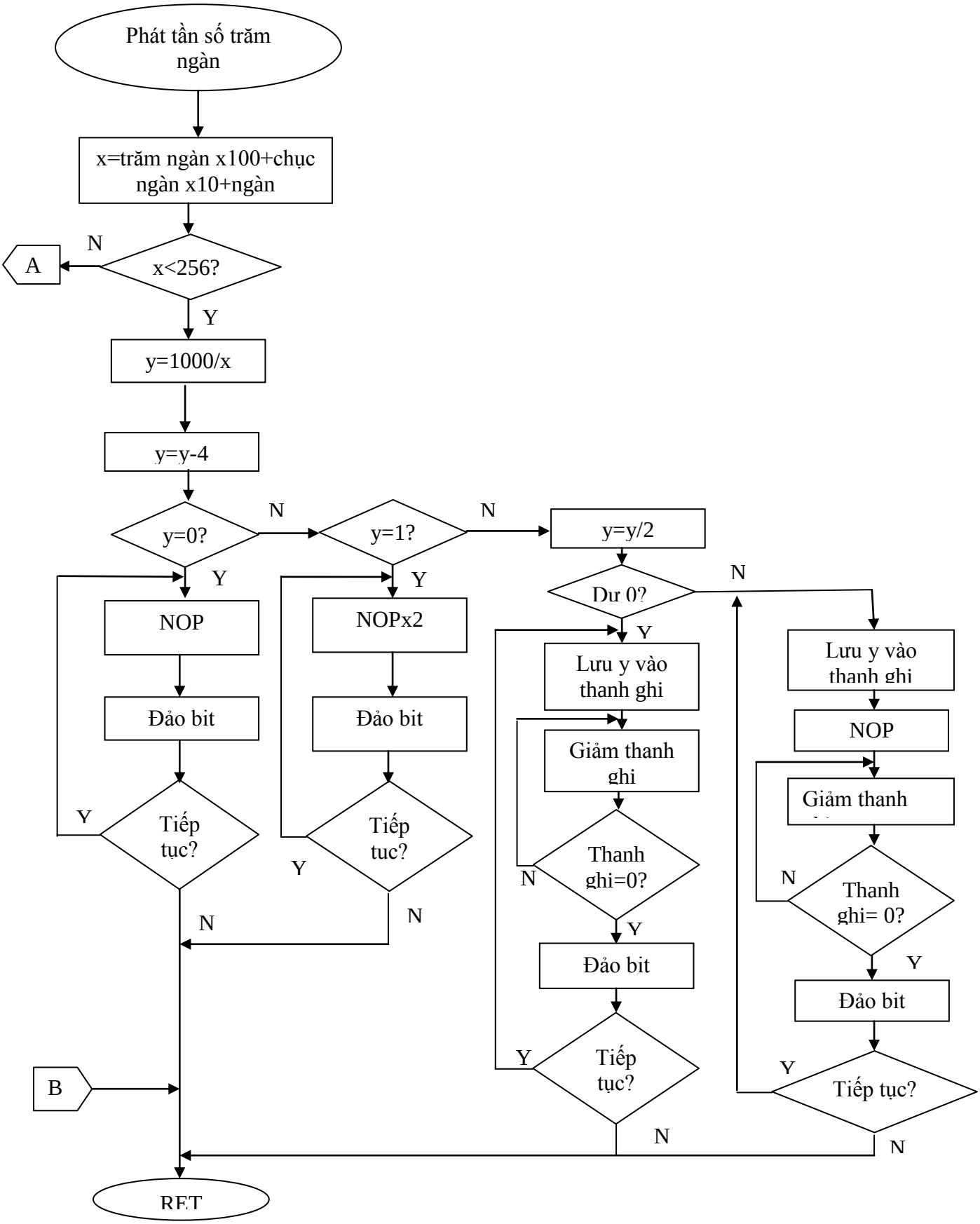
+ Chương trình con PC_LCD:

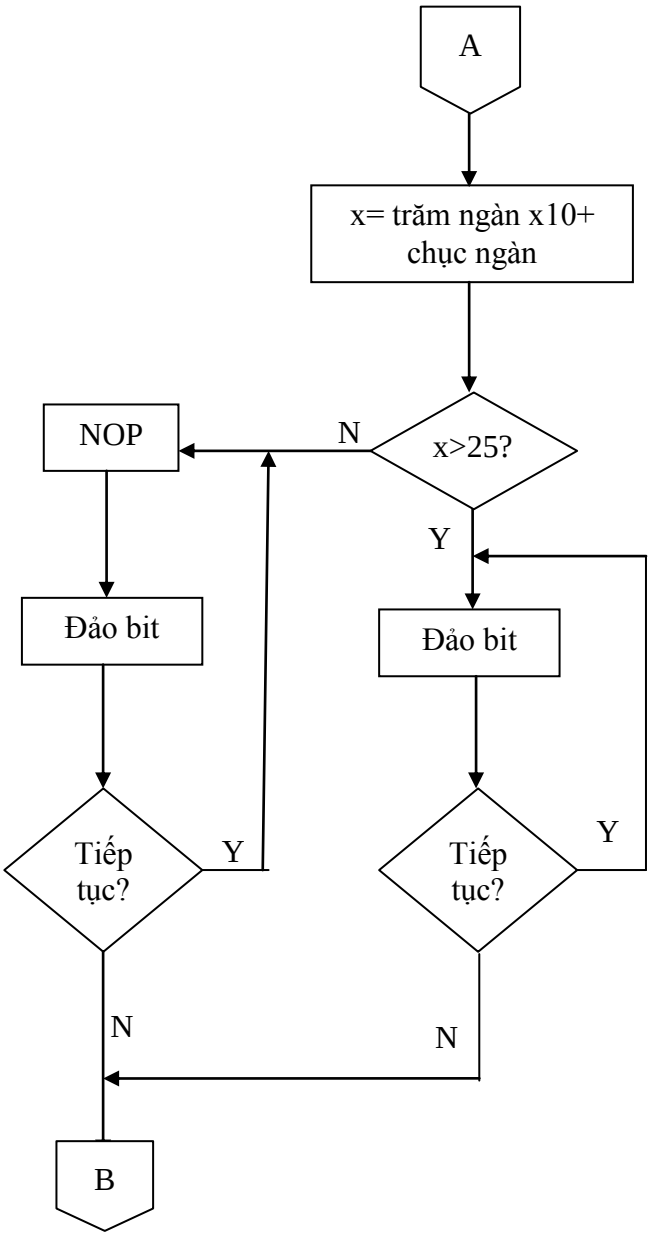


12.6 Lưu đồ thuật toán chương trình con PHÁT XUNG: (Tuấn + Somsamay)

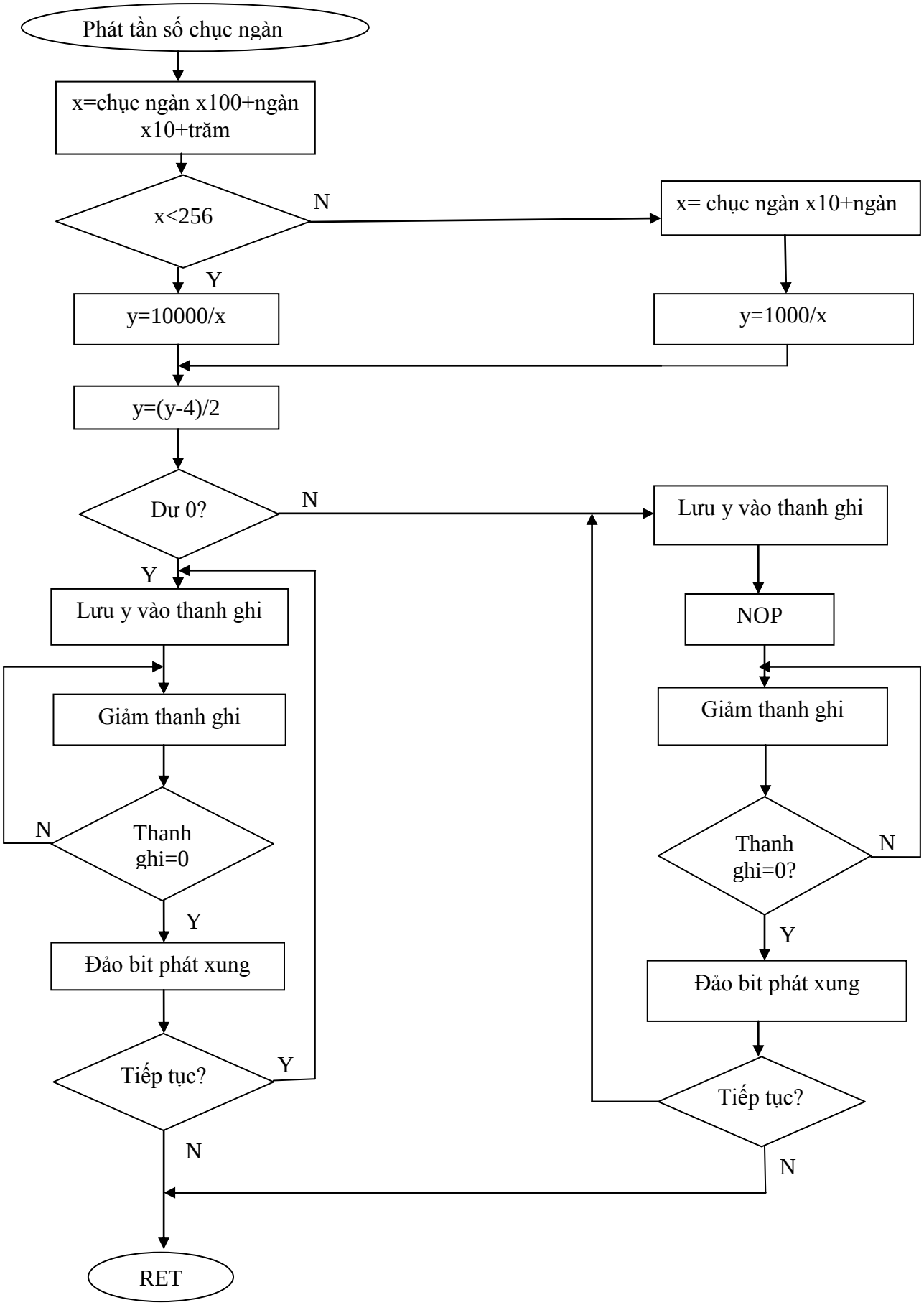


SƠ ĐỒ KHỐI CHƯƠNG TRÌNH CON PHÁT TẦN SỐ TRĂM NGÀN

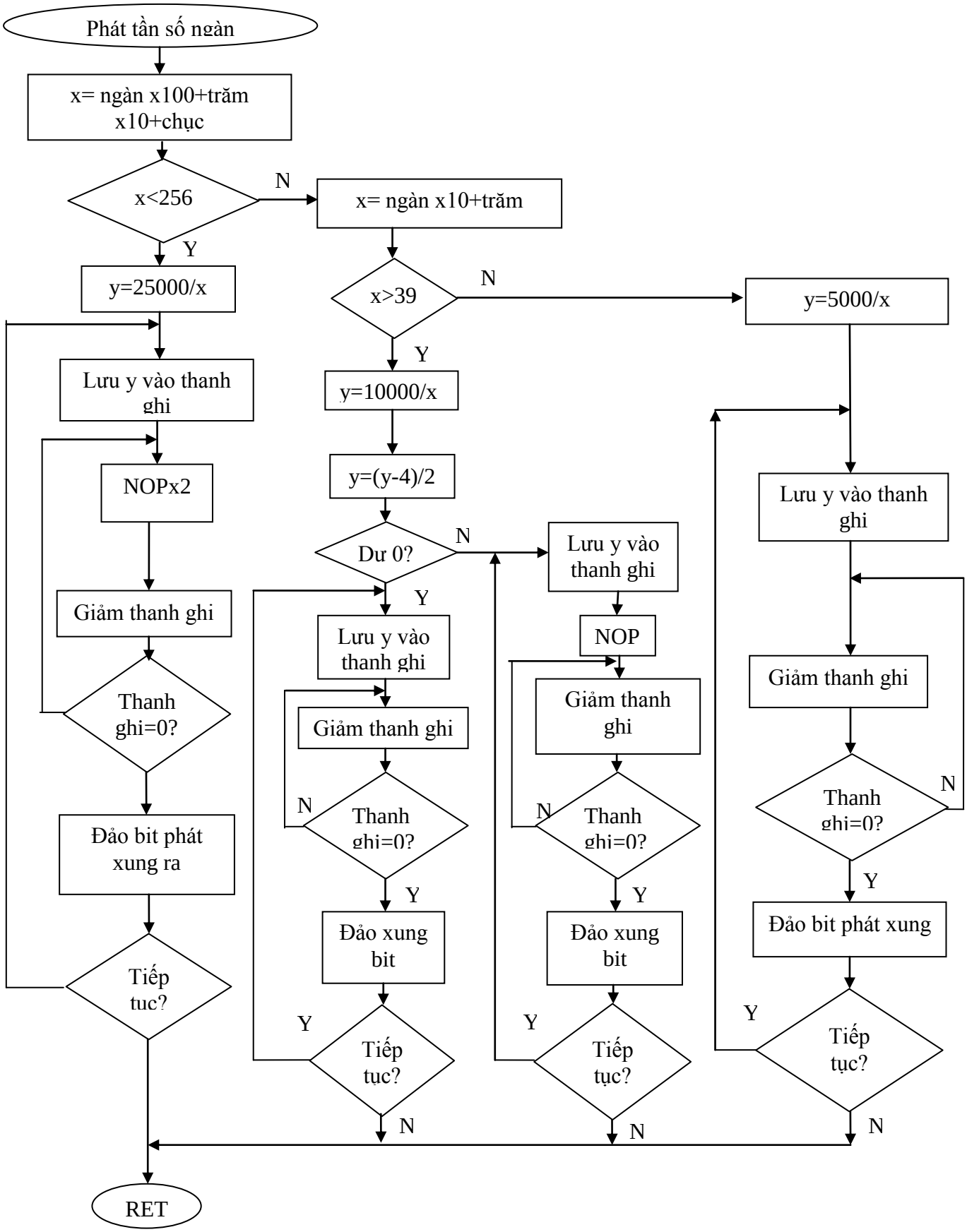



A
x= trăm ngàn x10+
chục ngàn
x>25?
N
NOP
Y
Đảo bit
Đảo bit
Tiếp
tục?
Y
Tiếp
tục?
Y
N
N
B

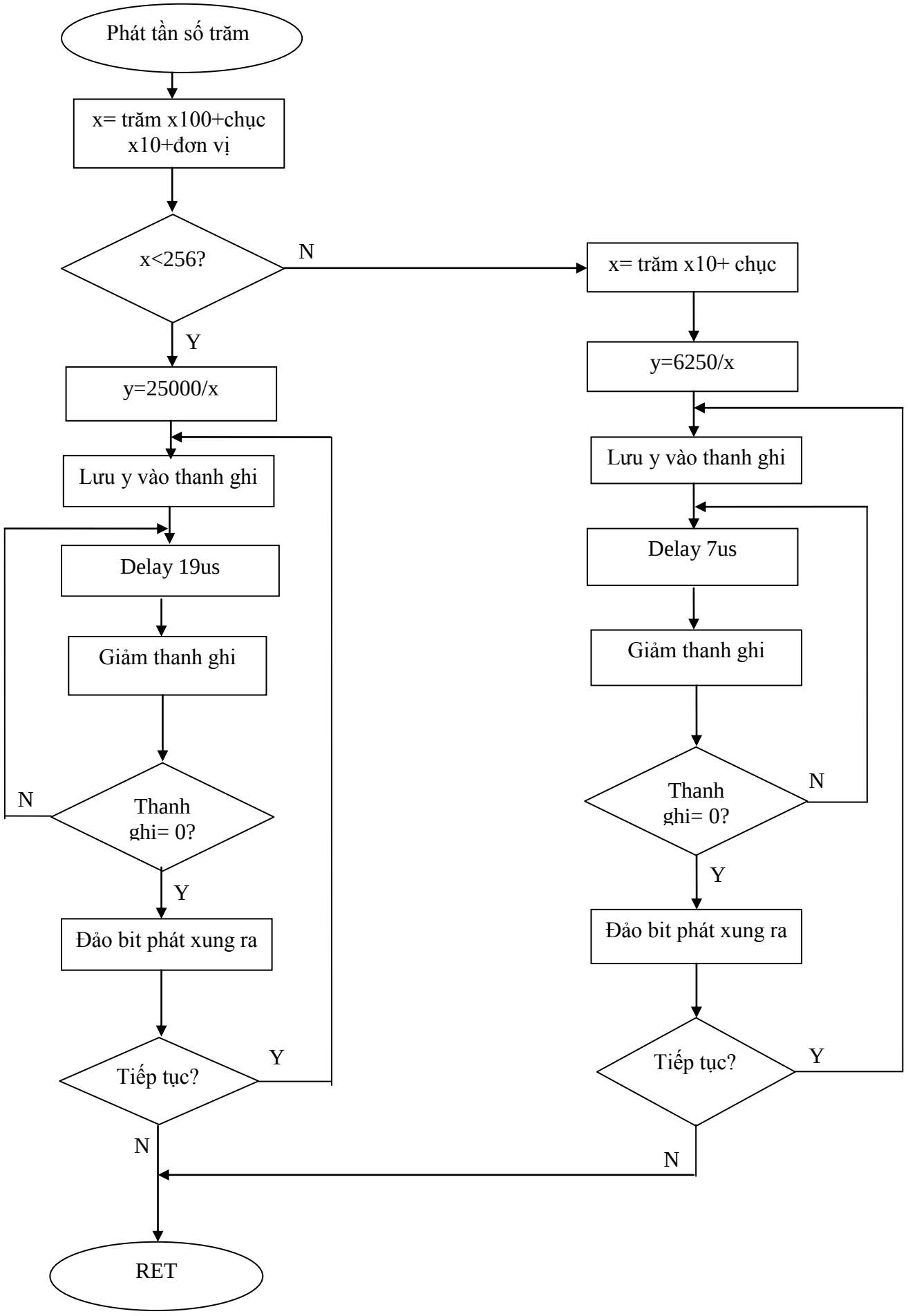
SƠ ĐỒ KHỐI CHƯƠNG TRÌNH CON PHÁT TẦN SỐ CHỤC NGÀN.



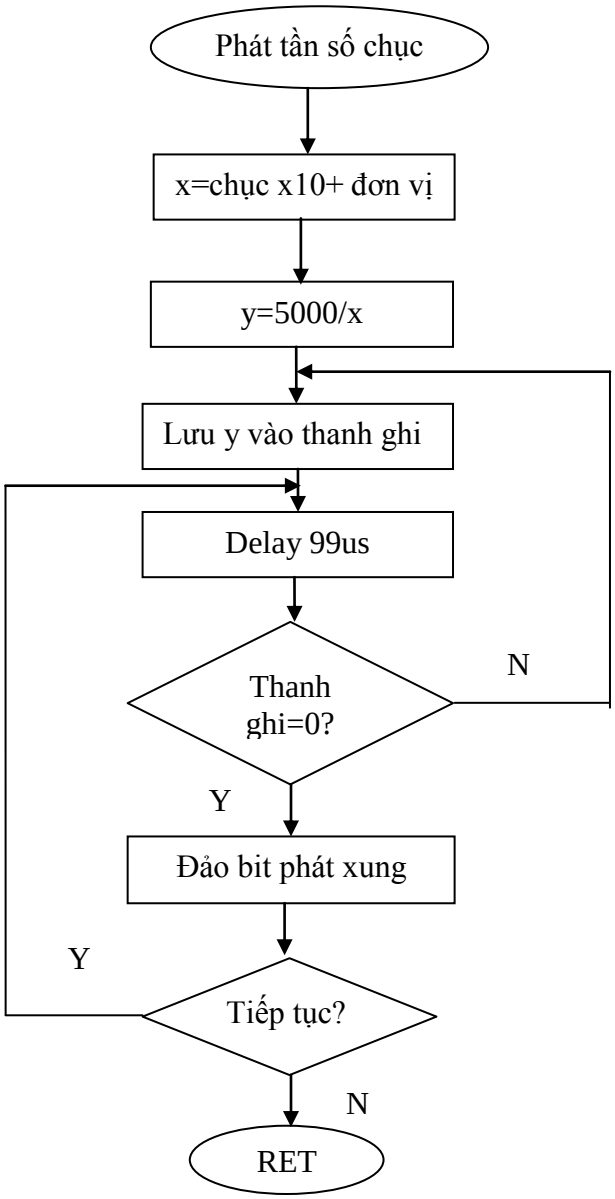
SƠ ĐỒ KHỐI CHƯƠNG TRÌNH CON PHÁT TẦN SỐ NGÀN



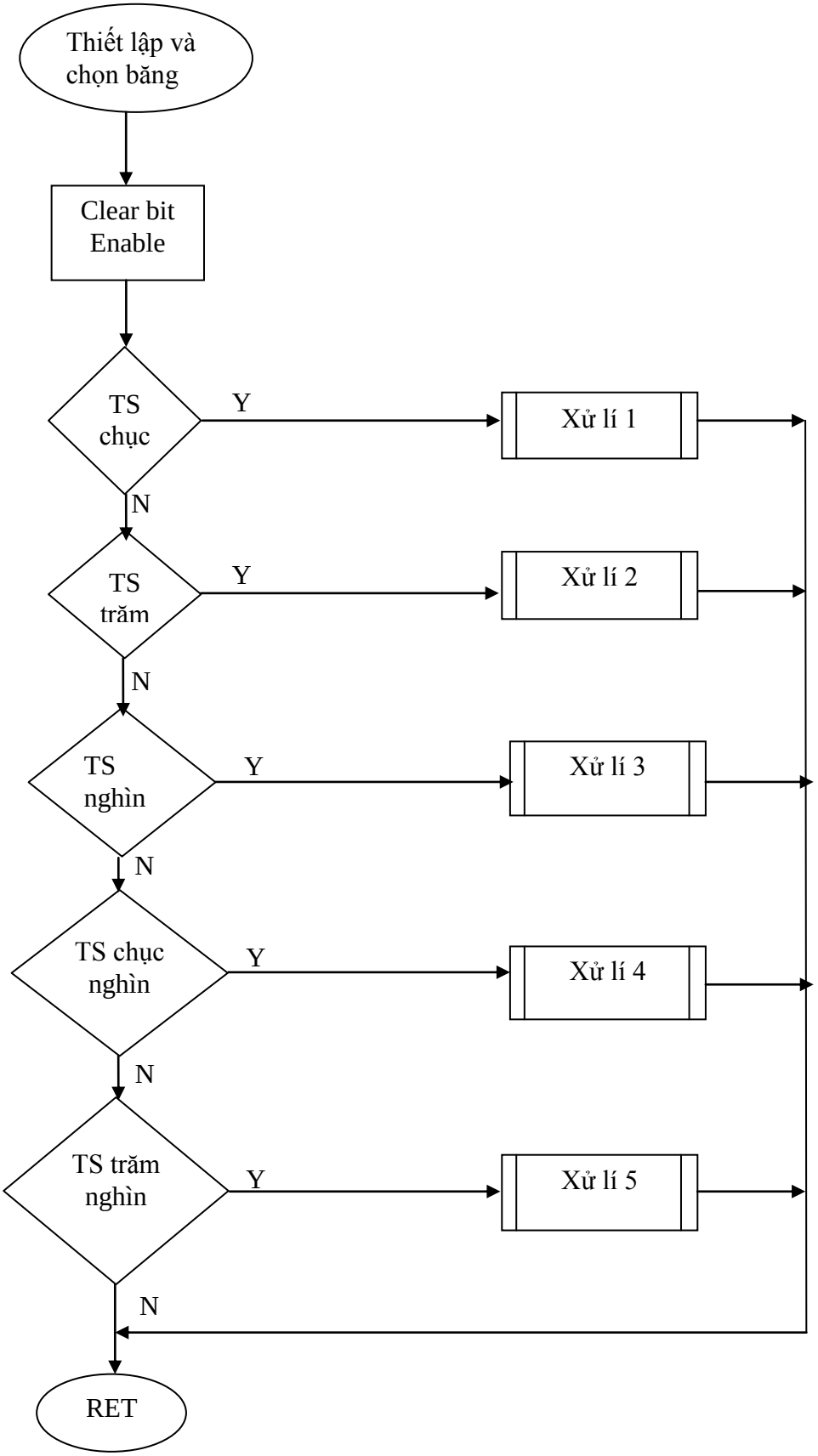
SƠ ĐỒ KHỐI CHƯƠNG TRÌNH CON PHÁT TẦN SỐ TRĂM

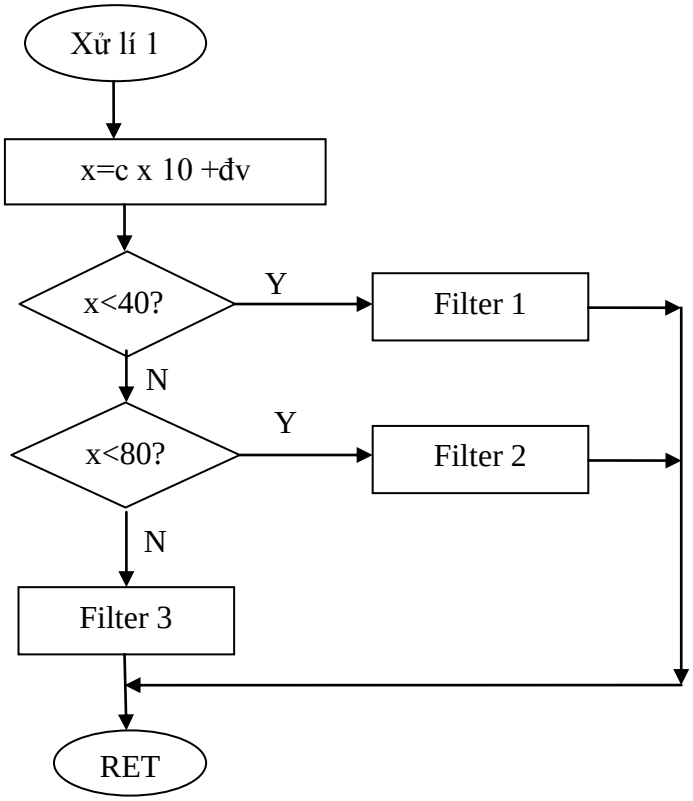


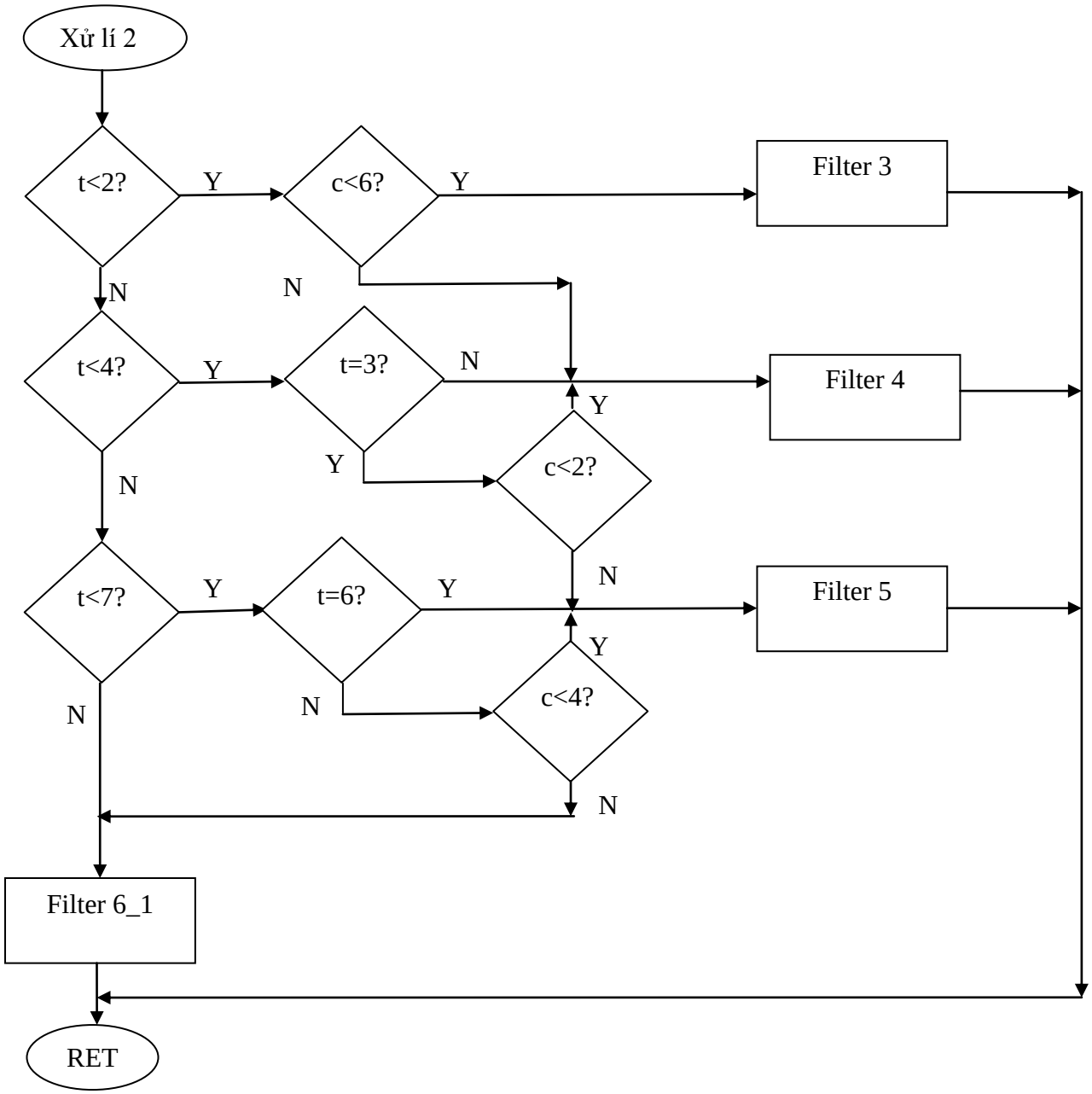
SƠ ĐỒ KHỐI CHƯƠNG TRÌNH CON PHÁT TẦN SỐ CHỤC

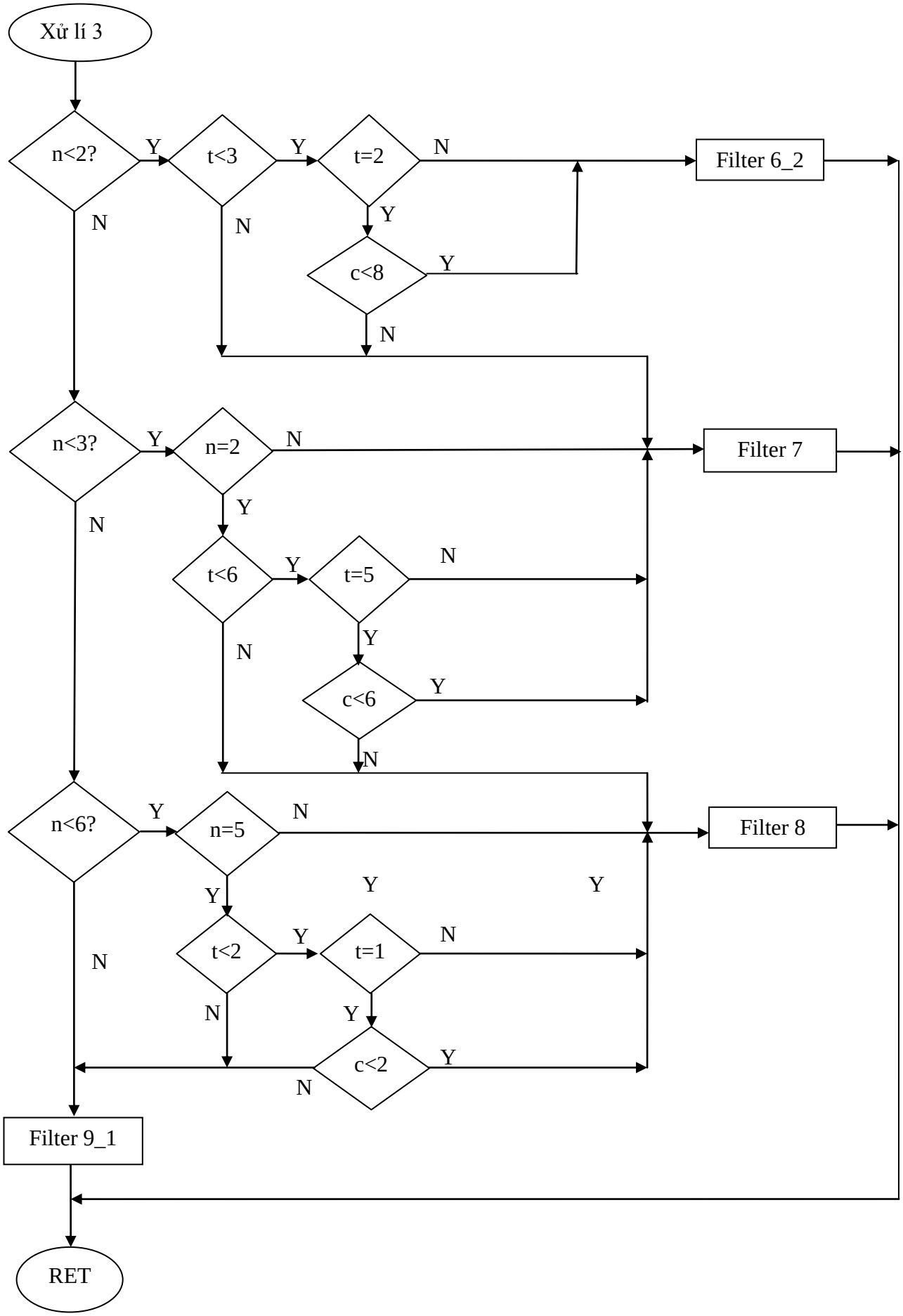


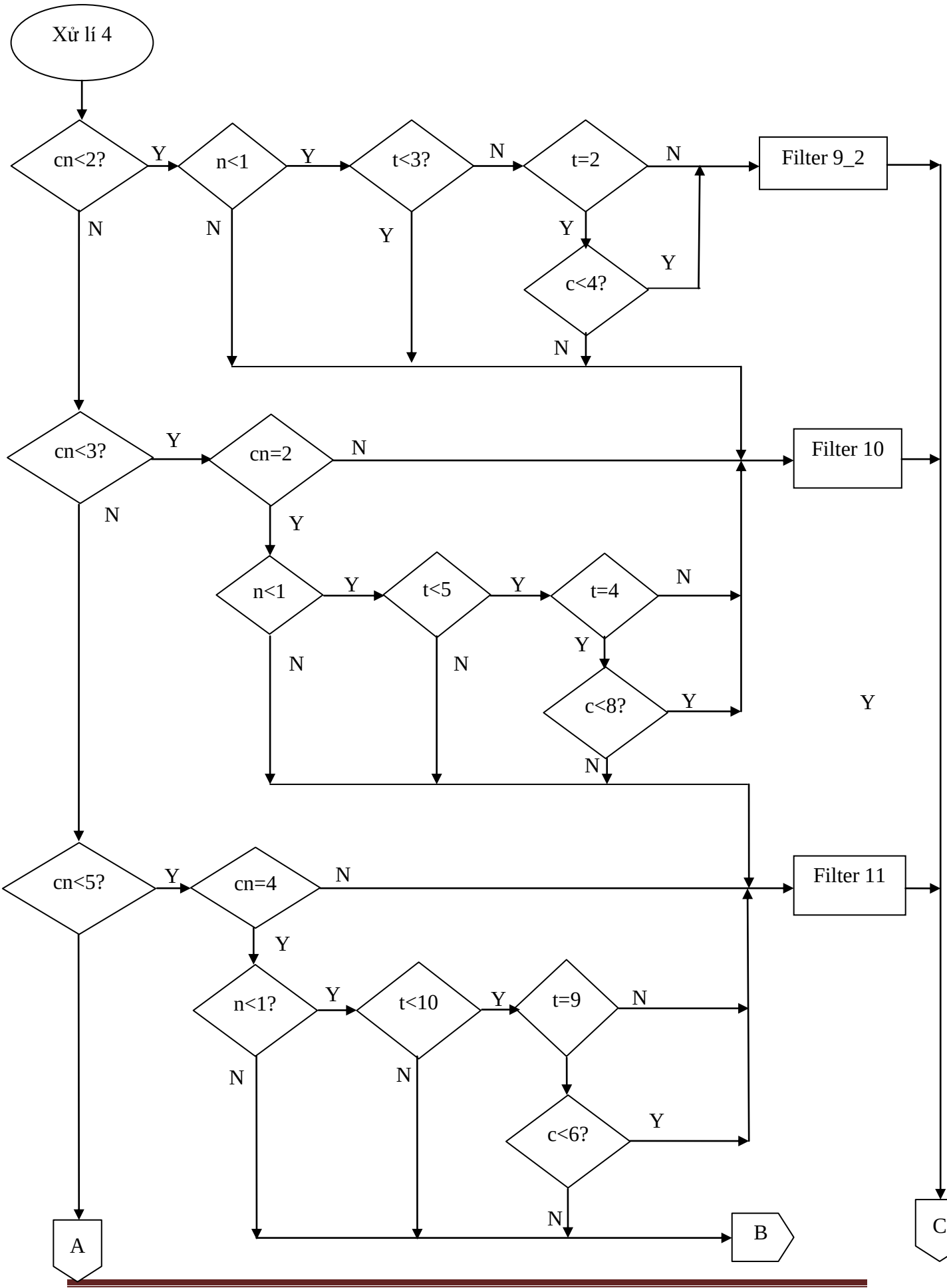
12.7 Lưu đồ thuật toán chương trình con THIẾT LẬP VÀ CHỌN BĂNG TẦN SỐ: (Kim Ngà)

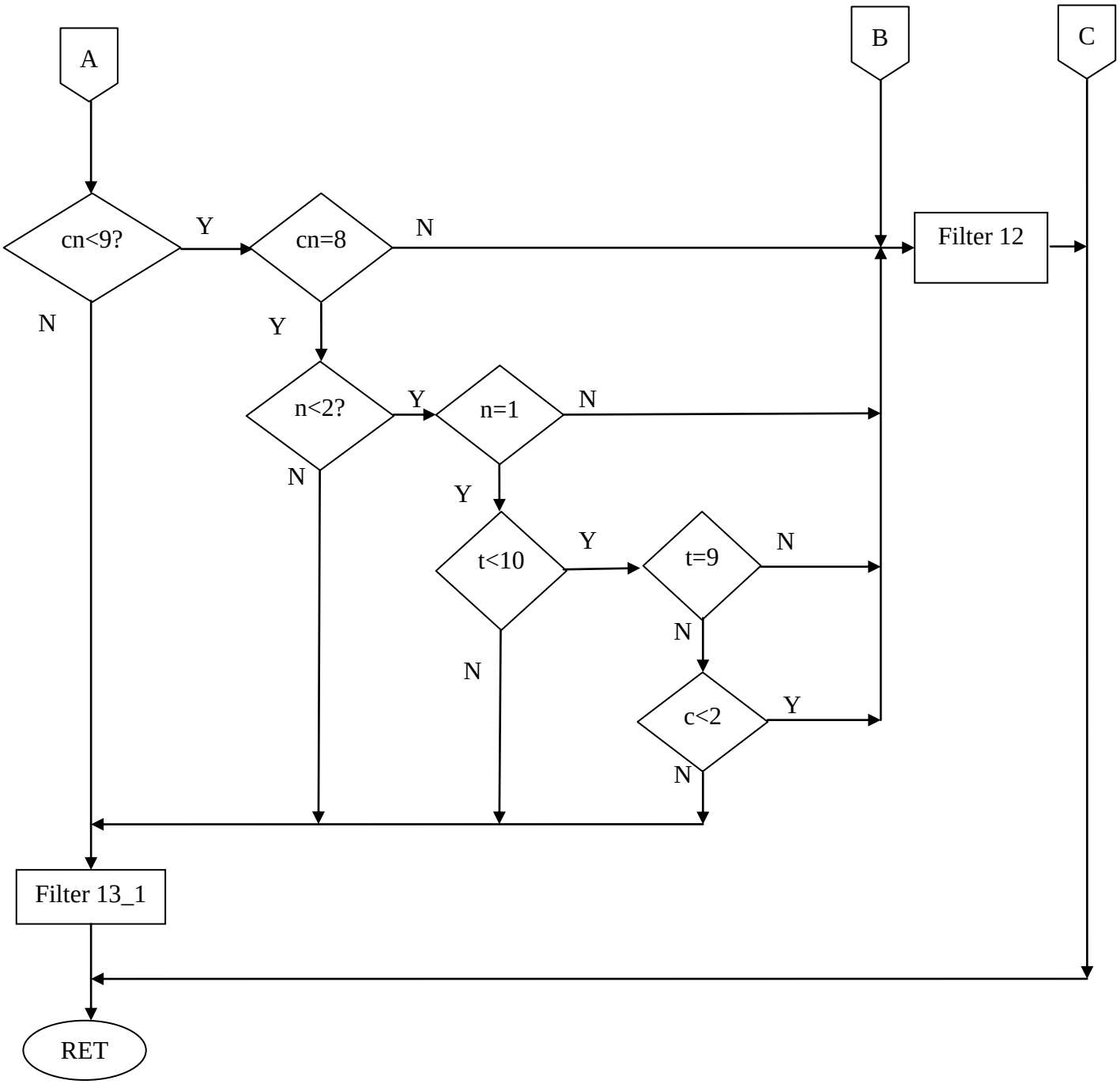


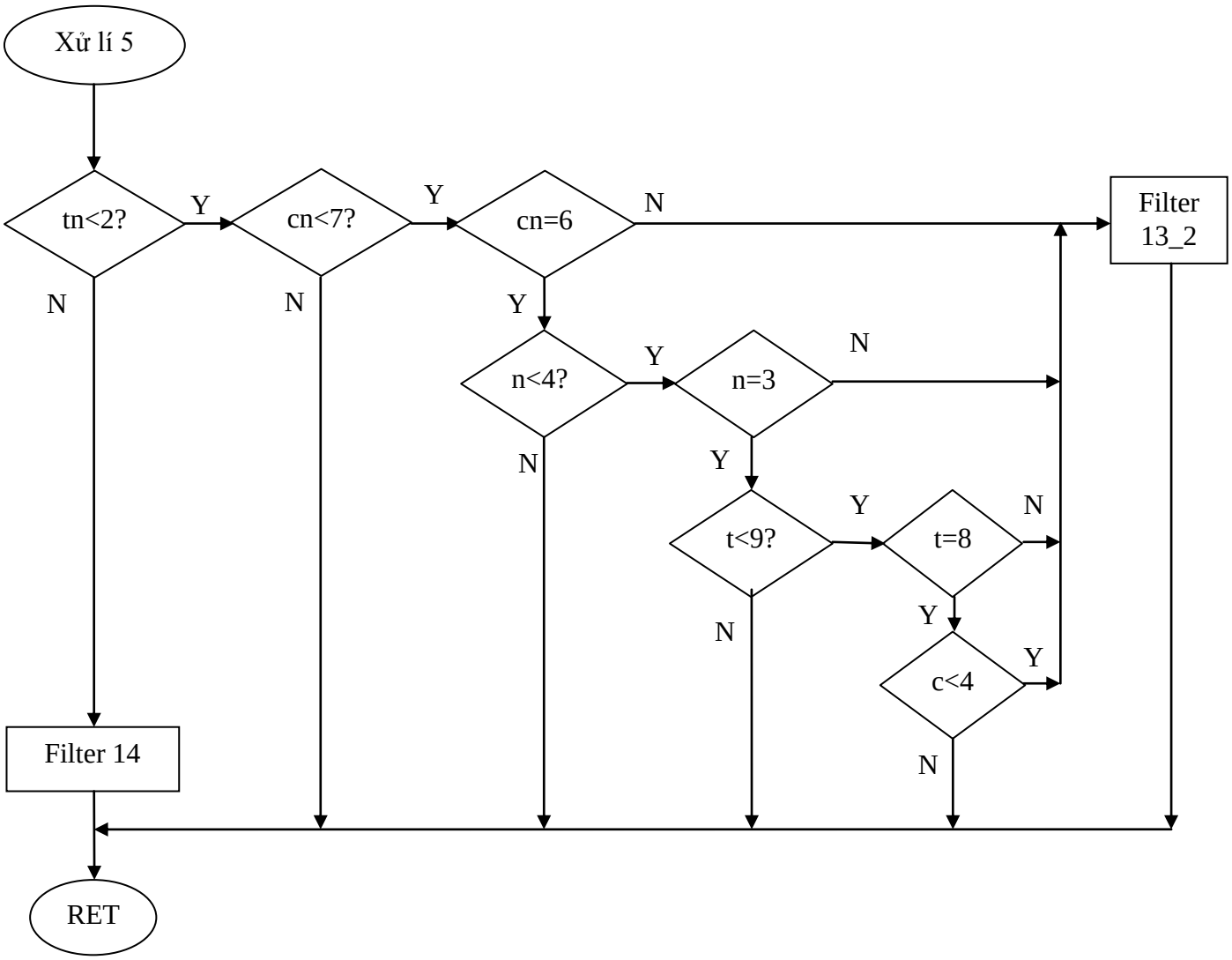

Xử lí 1
x=c x 10 +đv
x<40?
Y
Filter 1
N
x<80?
Y
Filter 2
N
Filter 3
RET


Xử lí 2
t<2?
Y
c<6?
Y
Filter 3
N
N
t<4?
Y
t=3?
N
Filter 4
Y
Y
c<2?
N
t<7?
Y
t=6?
Y
N
Filter 5
Y
N
c<4?
N
N
Filter 6_1
RET


Xử lí 3
n<2?
Y
t<3
Y
t=2
N
Filter 6_2
N
N
Y
c<8
Y
N
n<3?
Y
n=2
N
Filter 7
N
Y
t<6
Y
t=5
N
N
Y
c<6
Y
N
n<6?
Y
n=5
N
Filter 8
Y
Y
Y
t<2
Y
t=1
N
N
N
Y
c<2
Y
N
Filter 9_1
RET


Xử lí 4
cn<2?
Y
n<1
Y
t<3?
N
t=2
N
Filter 9_2
N
N
Y
Y
c<4?
Y
N
cn<3?
Y
cn=2
N
Filter 10
N
Y
n<1
Y
t<5
Y
t=4
N
N
N
Y
c<8?
Y
Y
N
cn<5?
Y
cn=4
N
Filter 11
Y
n<1?
Y
t<10
Y
t=9
N
N
N
c<6?
Y
N
A
B
C


A
B
C
cn<9?
Y
cn=8
N
Filter 12
N
Y
n<2?
Y
n=1
N
N
Y
t<10
Y
t=9
N
N
N
c<2
Y
N
Filter 13_1
RET


Xử lí 5
tn<2?
Y
cn<7?
Y
cn=6
N
Filter 13_2
N
N
Y
n<4?
Y
n=3
N
N
Y
t<9?
Y
t=8
N
N
Y
c<4
Y
N
Filter 14
RET

13. Viết chương trình:

13.1. Chương trình con thiết lập LCD: (Trâm)

```
LCD:
            LCALL       LCD_init
            LCALL       LCD_title
            RET
;=================KHOI_TAO=============
LCD_init:
            MOV     A,#38H   ;Function set: 2 Line, 8-bit, 5x7 dots
            LCALL   LCD_command
            MOV     A,#0EH        ;Display on, Cursor command
            LCALL   LCD_command
            MOV     A,#01H   ;Clear LCD
            LCALL   LCD_command
      RET
;=========HIEN THI TEN NHOM VA DE TAI============
LCD_title:
            MOV     DPTR,#Title
            MOV     R1,#11
back:       LCALL   LCD_sendstring
            LCALL   LCD_clear
            INC     DPTR
            DJNZ    R1,back
            RET
```

13.2. Chương trình con điều khiển CONTROL: (Trâm)

```
;=========KIEM TRA DIEU KHIEN BANG PC HAY KEYBOARD===========
;==========QUET PHIM=============================
CONTROL:
scan_key1:
        LCALL   IN_HEX
        JB      10,KEY_pc1  ; quet phim den khi co phim an
        SJMP     scan_key1
;================================================
KEY_pc1:
            CJNE        R6,#14,KEY_keyboard1
            CALL        PC_control ; Neu an nut 14 thi goi PC_control
            JMP         EXIT_CONTROL
KEY_keyboard1:
            CJNE        R6,#15,scan_key1
            CALL        KEYBOARD   ; Neu an nut 15 thi goi KEYBOARD
EXIT_CONTROL:
            RET
```

13.3 Chương trình con KEYBOARD: (Trâm)

```
KEYBOARD:
;==========CREATE RAM====================
            MOV     R0,#30H
            MOV     R1,#10
DEL:        MOV     @R0,#0          ; nap 0 vao vi tri duoc tro boi R0
            INC     R0
            DJNZ    R1,DEL
;=======Display f(Hz)= ==================
            MOV     DPTR,#Default
            LCALL   LCD_sendstring
            MOV     A,#86H   ; Place the cursor at the 7th position
of the first line
            LCALL   LCD_command
            MOV     R0,#30H
            MOV     R1,#0
```

```
;=========================================
scan_key:
        LCALL   IN_HEX
        JB      10,KEY_clear
        JMP     scan_key
;=======Press function buttons==============
KEY_clear:
        CJNE    R6,#10,KEY_start
        LCALL   LCD_clear
        CALL     KEYBOARD
KEY_start:
           CJNE       R6,#11,KEY_wave
           JB         BIT11,GuiDuLieu
           MOV        DPTR,#square
           ACALL      cursor
           CALL  D    ELAY
GuiDuLieu:
           LCALL      LCD_clear
; Display 'Data sent. Press TT to cont'
           MOV        DPTR,#sent
           LCALL      LCD_sendstring
           MOV         A,#0C0H   ; Place the cursor at the 1st
position of the SECOND line
           LCALL      LCD_command
           MOV        DPTR,#Continue
           LCALL      LCD_sendstring
           CALL       DATA_SENT
           CALL       INIT_UNIQUE1
           CALL       DANGSONG
; Check whether TT is pressed
CheckTT:
           LCALL      IN_HEX
           JB         10,KEY_cont1
           JMP        CheckTT
KEY_cont1:
           CJNE       R6,#13,CheckTT
           LCALL      LCD_clear
           LCALL      CONTROL
KEY_wave:             ; phim kiem tra dang song
           CJNE       R6,#12,KEY_cont2
           SETB       BIT11 ; neu nut12 an thi dang song sin
           MOV        DPTR,#sin
           LCALL      cursor
           JMP        scan_key
KEY_cont2:
           CJNE       R6,#13,KEY_pc2
           JMP        scan_key
KEY_pc2:
           CJNE       R6,#14,KEY_keyboard2
           JMP        scan_key
KEY_keyboard2:
           CJNE       R6,#15,CONT
           JMP        scan_key
cursor:    MOV        A,#0C0H    ; Place the cursor at the 1st
position of the SECOND line
           LCALL      LCD_command
           LCALL      LCD_sendstring
           MOV        A,R1
           ADD        A,#86H
           LCALL      LCD_command
```

```
           RET
;=======Press number buttons==============
CONT:      MOV        A,R6
           MOV        R5,A
rescan_key:
           LCALL      IN_HEX
           JB         10,rescan_key
           MOV        A,R5
;=============Save&Display Number==========
           MOV     @R0,A
           ADD      A,#48
           LCALL   LCD_senddata
           INC     R0
           INC     R1
           JMP    scan_key
           RET
;=========CAC CHUONG TRINH CON=============
;============BUSY_FLAG====================
LCD_busy:
           SETB   LCD_D7
           CLR    LCD_rs        ;RS=0, Select command register
           SETB   LCD_rw        ;RW=1, we are reading
check:
         CLR    LCD_en        ;Enable H->L
         SETB   LCD_en
         JB     LCD_D7,check;read busy flag again till it becomes 0
          RET                      ;Return from busy routine
;=============COMMAND=====================
LCD_command:
           LCALL   LCD_busy
           CLR     LCD_rs        ; RS=0, IR selected
           CLR     LCD_rw        ; RW=0, write to LCD
           MOV     LCD_data,A
           SETB    LCD_en
           CLR     LCD_en
           RET
;=============CLEAR LCD====================
LCD_clear:
           MOV     A,#01H        ;Clear LCD
           LCALL   LCD_command
           RET
;=============HIENTHI======================
LCD_senddata:
           LCALL   LCD_busy    ;Wait for LCD to process the data
           SETB    LCD_rs        ;RS=1, DR selected
           CLR     LCD_rw        ;RW=0, write to LCD
           MOV     LCD_data,A
           SETB    LCD_en
           CLR     LCD_en
           RET
;=============SEND STRING==================
LCD_sendstring:
           CLR        A                ;clear Accumulator for any
previous data
           MOVC       A,@A+DPTR        ;load the first character in
accumulator
           JZ         EXIT_send        ;go to exit if zero
           LCALL      LCD_senddata     ;send first char
           LCALL      DELAY             ;create delay time
           INC        DPTR             ;increment data pointer
```

```
            SJMP        LCD_sendstring   ;jump back to send the next
character
EXIT_send:
            RET
;================DELAY====================
DELAY:
            MOV   R5,#100
L3:         CALL  DELAYMS
            DJNZ  R5,L3
            RET
DELAYMS:                        ;Xtal=24MHz, Tm=0.5us
            MOV     R6,#3
L1:         MOV     R7,#250
L2:         DJNZ    R7,L2
            DJNZ    R6,L1
    RET
;==============TRA_BANG===============
 Title:
 DB         'HELLO',00H
 DB   'TO 1 - NHOM 9',00H
 DB   'NG THI BAO TRAM',00H
 DB   'NG LE HOANG TUAN',00H
 DB   'VO NGOC TRI MINH',00H
 DB   'TRAN PHUOC THINH',00H
 DB   'TRUONG KIM NGA',00H
 DB   'SENGSAY SOMSAMAY',00H
 DB   'MACH PHAT TAN SO',00H
 DB   'DOI TUONG DKHIEN',00H
 DB   '14.PC OR 15.KEY',00H
 Default:
 DB   'f(Hz)=',00H
 sin:
 DB   'waveform: sine',00H
 square:
 DB 'waveform: square',00H
 sent:
 DB 'Data sent',00H
 Continue:
 DB 'Press TT to continue',00H
```

Chương trình con quét phím: (Phước Thịnh)

```
;===========SCAN BAN PHIM==================
IN_HEX:
        MOV     R3,#70
BACK1:  LCALL   GET_KEY
        JNB     10,EXP1
        DJNZ    R3,BACK1
        MOV     R3,#50
BACK2:  LCALL   GET_KEY
        DJNZ    R3,BACK2
EXP1:   NOP
    RET
GET_KEY:
        MOV     A,#0FEH
        MOV     R2,#0
SCAN_ROW:
        MOV     P1,A
        MOV     R4,A
```

```
        JNB     P1.4,ROW_0
        JNB     P1.5,ROW_1
        JNB     P1.6,ROW_2
        JNB     P1.7,ROW_3
        MOV     A,R4
        RL      A
        INC     R2
        CJNE    R2,#4,SCAN_ROW
        SJMP    NO_CODE
ROW_0:  MOV     A,R2
        ADD     A,#0
        SETB    10
        MOV     R6,A
        SJMP    EXIT
ROW_1:  MOV     A,R2
        ADD     A,#4
        SETB    10
        MOV     R6,A
        SJMP    EXIT
 ROW_2: MOV     A,R2
        ADD     A,#8
        SETB    10
        MOV     R6,A
        SJMP    EXIT
 ROW_3: MOV     A,R2
        ADD     A,#12
        SETB    10
        MOV     R6,A
        SJMP    EXIT
NO_CODE:
        CLR 10
EXIT:
    RET
```

13.4 Chương trình con PC_control: (Trí Minh)

```
PC_control:
     CALL NHAN_DULIEU
      CALL PC_LCD
      CALL CHUYENDOI
      MOV R0,#30H
      CALL DANGSONG
  RET

PC_LCD:
           LCALL      LCD_clear
           MOV        DPTR,#Default
           LCALL      LCD_sendstring
           MOV        R0,#30H
           MOV        R1,#7
DisplayNumber:
           MOV        A,@R0
           ADD        A,#48
           LCALL      LCD_senddata
           LCALL      DELAY
           INC        R0
           DJNZ  R1, DisplayNumber
      RET
```

```
CHUYENDOI:
     MOV R0,#30H                  ; VUNG NHO CU
      CLR C                       ; XOA CO C
      MOV A,#7
      SUBB A,SAVE_COUNT           ; A=A-R1 VOI R1 LUU BIEN DEM SO
LUONG CAC SO NHAP VAO
      MOV R5,A                    ; R5=7-SAVE_COUNT
      MOV R1,#40H                      ; VUNG NHO MOI
STEP1:
      INC R0
      DJNZ R5,STEP1
      MOV R6,SAVE_COUNT
STEP2:
    MOV A,@R0
      MOV @R1,A
      INC R0
      INC R1
      DJNZ R6,STEP2
      MOV R1,#40H
      MOV R0,#30H
      MOV R6,SAVE_COUNT
STEP3:
    MOV A,@R1
      MOV @R0,A
      INC R0
      INC R1
      DJNZ R6,STEP3
  RET

NHAN_DULIEU:
    CALL WAVEFORM_GT         ; KIEM TRA DANG SONG GUI XUONG
      CJNE R0,#49H,VUONG_GT  ; R0=55H LA PHAT SONG VUONG
      CALL SIN_GT                      ; R0=49H LA PHAT SONG SIN
  RET
WAVEFORM_GT:                       ; LAY GIA TRI KIEU DANG SONG GUI
XUONG  VA SO LUONG KI TU GUI XUONG LUU VAO SAVE_COUNT
      JNB RI,$                     ; DOI KIEU DANG SONG GUI XUONG
      MOV A,SBUF
      CLR RI
      MOV R0,A

      JNB RI,$                     ; DOI SO LUONG KI TU GUI XUONG LUU
VAO SAVE_COUNT
      MOV A,SBUF
      CLR RI
      MOV R1,A
      MOV SAVE_COUNT,R1
  RET
VUONG_GT:
      CLR  BIT11                   ; NEU LA SONG VUONG THI CLEAR
BIT11=0
      CALL INIT_UNIQUE1            ; KHOI TAO VUNG LUU GIA TRI DU LIEU
GUI XUONG
      CALL CHO_NHAN
  RET
SIN_GT:
    SETB BIT11                     ; NEU LA SONG SIN THI SETB BIT11=1
      CALL INIT_UNIQUE1            ; KHOI TAO VUNG LUU GIA TRI DU LIEU
GUI XUONG
```

```
      CALL CHO_NHAN
  RET

/**************LUU DU LIEU TAN SO CAN PHAT TU PC XUONG VDK TAI VUNG
NHO TU 30H DEN 36H***********/
CHO_NHAN:                    ; CHO` NHAN DU LIEU GUI XUONG TIEP TUC
TU PC
    JNB RI,$
      MOV A,SBUF
      CLR RI
      MOV @R0,A
      INC R0
      DJNZ R5,CHO_NHAN
  RET
```

13.5 Chương trình phát xung: (Hoàng Tuấn+ Somsamay)

```
;===========CT_CON_PHATXUNG============

PHATXUNG:
            CJNE R5,#2,PHAT1
            CALL TSCHUC
            JMP EXIT_PX
      PHAT1:
            CJNE R5,#3,PHAT2
            CALL TSTRAM
            JMP EXIT_PX
      PHAT2:
            CJNE R5,#4,PHAT3
            CALL TSNGAN
            JMP EXIT_PX
      PHAT3:
            CJNE R5,#5,PHAT4
            CALL TSCHUCNGAN
            JMP EXIT_PX
      PHAT4:
            CALL TSTRAMNGAN
      EXIT_PX:
            RET

;===========CT_CON_TSCHUC=============

TSCHUC:
            CALL TINH3
      CHUC1:
            CALL TINH1
            MOV R6,A
            SUBB A,#13H
            JC CHUC1
            CJNE R6,#13H,CHUC2
            MOV A,R1
            SUBB A,#89H
            JC CHUC1
      CHUC2:
            DEC R4
            MOV A,R4
      CHUC3:
            MOV R5,A
      CHUC4:
```

```
            CALL DELAY99US
            DJNZ R5,CHUC4
            CPL P3.7
            JMP  CHUC3
            RET

;===========CT_CON_TSTRAM============

TSTRAM:
            CLR C
            MOV R0,#30H
            MOV A,@R0
            SUBB A,#3
            JNC TRAM6
            CJNE @R0,#2,TRAM1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC TRAM6
            CJNE @R0,#5,TRAM1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC TRAM6
      TRAM1:
            CALL TINH4
      TRAM2:
            CALL TINH1
            MOV R6,A
            SUBB A,#61H
            JC TRAM2
            CJNE R6,#61H,TRAM3
            MOV A,R1
            SUBB A,#0A9H
            JC TRAM2
      TRAM3:
            DEC R4
            MOV A,R4
      TRAM4:
            MOV R5,A
      TRAM5:
            CALL DELAY19US
            DJNZ R5,TRAM5
            CPL P3.7
            SJMP TRAM4
            JMP EXIT_TRAM
      TRAM6:
            CALL TINH3
      TRAM7:
            CALL TINH1
            MOV R6,A
            SUBB A,#18H
            JC TRAM7
            CJNE R6,#18H,TRAM8
            MOV A,R1
            SUBB A,#06BH
            JC TRAM7
      TRAM8:
            DEC R4
            MOV A,R4
```

```
      TRAM9:
            MOV R5,A
      TRAM10:
            CALL DELAY7US
            DJNZ R5,TRAM10
            CPL P3.7
            SJMP TRAM9
      EXIT_TRAM:
            RET

;===========CT_CON_TSNGAN============

TSNGAN:
            CLR C
            MOV R0,#30H
            MOV A,@R0
            SUBB A,#3
            JNC NGAN6
            CJNE @R0,#2,NGAN1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC NGAN6
            CJNE @R0,#5,NGAN1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC NGAN6
      NGAN1:
            CALL TINH4
      NGAN2:
            CALL TINH1
            MOV R6,A
            SUBB A,#61H
            JC NGAN2
            CJNE R6,#61H,NGAN3
            MOV A,R1
            SUBB A,#0A9H
            JC NGAN2
      NGAN3:
            DEC R4
            MOV A,R4
      NGAN4:
            MOV R5,A
      NGAN5:
            NOP
            NOP
            DJNZ R5,NGAN5
            CPL P3.7
            SJMP NGAN4
            JMP EXIT_NGAN
      NGAN6:
            CALL TINH3
            SUBB A,#40
            JC NGAN13
      NGAN7:
            CALL TINH1
            MOV R6,A
            SUBB A,#27H
```

```
            JC NGAN7
            CJNE R6,#27H,NGAN8
            MOV A,R1
            SUBB A,#11H
            JC NGAN7
      NGAN8:
            MOV A,R4
            SUBB A,#5
            MOV R4,A
            CALL TINH2
            CJNE A,#0,NGAN11
            MOV A,R4
      NGAN9:
            MOV R5,A
      NGAN10:
            DJNZ R5,NGAN10
            CPL P3.7
            SJMP NGAN9
            JMP EXIT_NGAN
      NGAN11:
            MOV A,R4
            MOV R5,A
      NGAN12:
            DJNZ R5,NGAN12
            CPL P3.7
            SJMP NGAN11
            JMP EXIT_NGAN
      NGAN13:
            CALL TINH1
            MOV R6,A
            SUBB A,#13H
            JC NGAN13
            CJNE R6,#13H,NGAN14
            MOV A,R1
            SUBB A,#89H
            JC NGAN13
      NGAN14:
            DEC R4
            MOV A,R4
      NGAN15:
            MOV R5,A
      NGAN16:
            DJNZ R5,NGAN16
            CPL P3.7
            SJMP NGAN15
      EXIT_NGAN:
            RET

;==========CT_CON_TSCHUCNGAN==========

TSCHUCNGAN:
            CLR C
            MOV R0,#30H
            MOV A,@R0
            SUBB A,#3
            JNC CNGAN8
            CJNE @R0,#2,CNGAN1
            INC R0
            MOV A,@R0
            SUBB A,#6
```

```
            JNC CNGAN8
            CJNE @R0,#5,CNGAN1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC CNGAN8
      CNGAN1:
            CALL TINH4
      CNGAN2:
            CALL TINH1
            MOV R6,A
            SUBB A,#27H
            JC CNGAN2
            CJNE R6,#27H,CNGAN3
            MOV A,R1
            SUBB A,#11H
            JC CNGAN2
      CNGAN3:
            MOV A,R4
            SUBB A,#5
            MOV R4,A
            CALL TINH2
            CJNE A,#0,CNGAN6
            MOV A,R4
      CNGAN4:
            MOV R5,A
      CNGAN5:
            DJNZ R5,CNGAN5
            CPL P3.7
            SJMP CNGAN4
            JMP EXIT_CNGAN
      CNGAN6:
            MOV A,R4
            MOV R5,A
      CNGAN7:
            DJNZ R5,CNGAN7
            CPL P3.7
            SJMP CNGAN6
            JMP EXIT_CNGAN
      CNGAN8:
            CALL TINH3
      CNGAN9:
            CALL TINH1
            MOV R6,A
            SUBB A,#03H
            JC CNGAN9
            CJNE R6,#03H,CNGAN3
            MOV A,R1
            SUBB A,#0E9H
            JC CNGAN9
            JMP CNGAN3
      EXIT_CNGAN:
            RET

;==========CT_CON_TSTRAMNGAN==========

TSTRAMNGAN:
            CLR C
            MOV R0,#30H
            MOV A,@R0
```

```
      SUBB A,#3
      JNC TNGAN12
      CJNE @R0,#2,TNGAN1
      INC R0
      MOV A,@R0
      SUBB A,#6
      JNC TNGAN12
      CJNE @R0,#5,TNGAN1
      INC R0
      MOV A,@R0
      SUBB A,#6
      JNC TNGAN12
TNGAN1:
      CALL TINH4
TNGAN2:
      CALL TINH1
      MOV R6,A
      SUBB A,#03H
      JC TNGAN2
      CJNE R6,#03H,TNGAN3
      MOV A,R1
      SUBB A,#0E9H
      JC TNGAN2
TNGAN3:
      MOV A,R4
      SUBB A,#5
      MOV R4,A
      CJNE R4,#0,TNGAN5
TNGAN4:
      NOP
      CPL P3.7
      SJMP TNGAN4
      JMP EXIT_TNGAN
TNGAN5:
      CJNE R4,#1,TNGAN7
TNGAN6:
      NOP
      NOP
      CPL P3.7
      SJMP TNGAN6
      JMP EXIT_TNGAN
TNGAN7:
      CALL TINH2
      CJNE A,#0,TNGAN10
      MOV A,R4
TNGAN8:
      MOV R5,A
TNGAN9:
      DJNZ R5,TNGAN9
      CPL P3.7
      SJMP TNGAN8
      JMP EXIT_TNGAN
TNGAN10:
      MOV A,R4
      MOV R5,A
TNGAN11:
      DJNZ R5,TNGAN11
      CPL P3.7
      SJMP TNGAN10
      JMP EXIT_TNGAN
```

```
      TNGAN12:
            CALL TINH3
            SUBB A,#26
            JC TNGAN14
      TNGAN13:
            CPL P3.7
            SJMP TNGAN13
            JMP EXIT_TNGAN
      TNGAN14:
            NOP
            CPL P3.7
            SJMP TNGAN14
      EXIT_TNGAN:
            RET

;===========CT_CON_TINH1============

TINH1:
            CLR C
            INC R4
            MOV B,R4
            MOV A,R3
            MUL AB
            MOV R1,A
            MOV A,B
            RET

;===========CT_CON_TINH2============

TINH2:
            MOV A,R4
            MOV B,#2
            DIV AB
            MOV R4,A
            MOV A,B
            RET

;===========CT_CON_TINH3============

TINH3:
            MOV R0,#30H
            MOV A,@R0
            MOV B,#10
            MUL  AB
            INC R0
            ADD A,@R0
            MOV R3,A
            MOV R4,#0
            RET

;===========CT_CON_TINH4============


TINH4:
            MOV R0,#30H
            MOV A,@R0
            MOV B,#10
            MUL AB
            INC R0
            ADD A,@R0
```

```
            MOV B,#10
            MUL AB
            INC R0
            ADD A,@R0
            MOV R3,A
            MOV R4,#0
            RET
;==========CT_CON_DELAY99US==========

DELAY99US:
            NOP
            MOV R6,#96
      DL99:
            DJNZ  R6,DL99
            RET

;==========CT_CON_DELAY19US==========

DELAY19US:
            NOP
            MOV R6,#16
      DL19:
            DJNZ R6,DL19
            RET

;==========CT_CON_DELAY7US==========

DELAY7US:
            NOP
            MOV R6,#4
      DL7:
            DJNZ R6,DL7
            RET
```

13.6 Chương trình thiết lập và chọn băng: (Kim Ngà)

```
;==========CT_CON_CHONBANG==========

CHONBANG:
            CLR EN
            CJNE R5,#2,CHON1
            CALL XULY1
            JMP EXIT_CB
      CHON1:
            CJNE R5,#3,CHON2
            CALL XULY2
            JMP EXIT_CB
      CHON2:
            CJNE R5,#4,CHON3
            CALL XULY3
            JMP EXIT_CB
      CHON3:
            CJNE R5,#5,CHON4
            CALL XULY4
            JMP EXIT_CB
      CHON4:
            CALL XULY5
      EXIT_CB:
            RET
```

```
;==========CT_CON_XULY1==========

XULY1:
            MOV R0,#20H
            MOV A,@R0
            MOV B,#10
            MUL AB
            INC R0
            ADD A,@R0
            MOV R1,A
            SUBB A,#40
            JNC XL1_1
      FILTER1:
            CLR DC3
            CLR DC2
            CLR DC1
            CLR DC0
            JMP EXIT1
      XL1_1:
            MOV A,R1
            SUBB A,#80
            JNC FILTER3_1
      FILTER2:
            CLR DC3
            CLR DC2
            CLR DC1
            SETB DC0
            JMP EXIT1
      FILTER3_1:
            CLR DC3
            CLR DC2
            SETB DC1
            CLR DC0
      EXIT1:
            RET

;==========CT_CON_XULY2==========

XULY2:
            MOV R0,#20H
            MOV A,@R0
            SUBB A,#2
            JNC XL2_1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC FILTER4
      FILTER3_2:
            CLR DC3
            CLR DC2
            SETB DC1
            CLR DC0
            JMP EXIT2
      XL2_1:
            MOV R0,#20H
            MOV A,@R0
            SUBB A,#4
            JNC XL2_2
            CJNE @R0,#3,FILTER4
```

```
            INC R0
            MOV A,@R0
            SUBB A,#2
            JNC  FILTER5
      FILTER4:
            CLR DC3
            CLR DC2
            SETB DC1
            SETB DC0
            JMP EXIT2
      XL2_2:
            MOV R0,#20H
            MOV A,@R0
            SUBB A,#7
            JNC FILTER6_1
            CJNE @R0,#6,FILTER5
            INC R0
            MOV A,@R0
            SUBB A,#4
            JNC FILTER6_1
      FILTER5:
            CLR DC3
            SETB DC2
            CLR DC1
            CLR DC0
            JMP EXIT2
      FILTER6_1:
            CLR DC3
            SETB DC2
            CLR DC1
            SETB DC0

      EXIT2:
            RET

;==========CT_CON_XULY3==========


XULY3:
                  MOV R0,#20H
                  MOV A,@R0
                  SUBB A,#2
                        JNC XL3_1
                  INC R0
                  MOV A,@R0
                  SUBB A,#3
                  JNC FILTER7
                  CJNE @R0,#2,FILTER6_2
                  INC R0
                  MOV A,@R0
                  SUBB A,#8
                  JNC FILTER7
            FILTER6_2:
                  CLR DC3
                  SETB DC2
                  CLR DC1
                        SETB DC0
                        JMP EXIT3

XL3_1:
```

```
                MOV R0,#20H
                MOV A,@R0
                SUBB A,#3
                      JNC XL3_2
                CJNE @R0,#2,FILTER7
                INC R0
                MOV A,@R0
                SUBB A,#6
                JNC FILTER8
                CJNE @R0,#5,FILTER7
                INC R0
                MOV A,@R0
                SUBB A,#6
                JNC FILTER8
          FILTER7:
                CLR DC3
                SETB DC2
                SETB DC1
                      CLR DC0
                      JMP EXIT3
XL3_2:
                MOV R0,#20H
                MOV A,@R0
                SUBB A,#6
                      JNC FILTER9_1
                CJNE @R0,#5,FILTER8
                INC R0
                MOV A,@R0
                SUBB A,#2
                JNC FILTER9_1
                CJNE @R0,#1,FILTER8
                INC R0
                MOV A,@R0
                SUBB A,#2
                JNC FILTER9_1
          FILTER8:
                CLR DC3
                SETB DC2
                SETB DC1
                      SETB DC0
                      JMP EXIT3
          FILTER9_1:
                SETB DC3
                CLR DC2
                CLR DC1
                      CLR DC0
          EXIT3:
                RET

;==========CT_CON_XULY4==========

XULY4:
                MOV R0,#20H
                MOV A,@R0
                SUBB A,#2
                JNC XL4_1
                INC R0
                MOV A,@R0
                SUBB A,#1
                JNC FILTER10
```

```
                INC R0
                MOV A,@R0
                SUBB A,#3
                JNC FILTER10
                CJNE @R0,#2,FILTER9_2
                INC R0
                MOV A,@R0
                SUBB A,#4
                JNC FILTER10
          FILTER9_2:
                     SETB DC3
                CLR DC2
                CLR DC1
                     CLR DC0
                     JMP EXIT4

XL4_1:
                MOV R0,#20H
                MOV A,@R0
                SUBB A,#3
                JNC XL4_2
                CJNE @R0,#2,FILTER10
                INC R0
                MOV A,@R0
                SUBB A,#1
                JNC FILTER11
                INC R0
                MOV A,@R0
                SUBB A,#5
                JNC FILTER11
                CJNE @R0,#4,FILTER10
                INC R0
                MOV A,@R0
                SUBB A,#8
                JNC FILTER11
          FILTER10:
                     SETB DC3
                CLR DC2
                CLR DC1
                     SETB DC0
                     JMP EXIT4
XL4_2:
                MOV R0,#20H
                MOV A,@R0
                SUBB A,#5
                JNC XL4_3
                CJNE @R0,#4,FILTER11
                INC R0
                MOV A,@R0
                SUBB A,#1
                JNC FILTER12
                INC R0
                MOV A,@R0
                SUBB A,#10
                JNC FILTER12
                CJNE @R0,#9,FILTER11
                INC R0
                MOV A,@R0
                SUBB A,#6
                JNC FILTER12
```

```
            FILTER11:
                        SETB DC3
                  CLR DC2
                  SETB DC1
                        CLR DC0
                        JMP EXIT4
XL4_3:
                  MOV R0,#20H
                  MOV A,@R0
                  SUBB A,#9
                  JNC FILTER13_1
                  CJNE @R0,#8,FILTER12
                  INC R0
                  MOV A,@R0
                  SUBB A,#2
                  JNC FILTER13_1
                  CJNE @R0,#1,FILTER12
                  INC R0
                  MOV A,@R0
                  SUBB A,#10
                  JNC FILTER13_1
                  CJNE @R0,#9,FILTER12
                  INC R0
                  MOV A,@R0
                  SUBB A,#2
                  JNC FILTER13_1
            FILTER12:
                  SETB DC3
                  CLR DC2
                  SETB DC1
                        SETB DC0
                        JMP EXIT4
            FILTER13_1:
                        SETB DC3
                  SETB DC2
                  CLR DC1
                        CLR DC0
            EXIT4:
                  RET

;==========CT_CON_XULY5==========

XULY5:
                  MOV R0,#20H
                  MOV A,@R0
                  SUBB A,#2
                  JNC FILTER14
                  INC R0
                  MOV A,@R0
                  SUBB A,#7
                  JNC FILTER14
                  CJNE @R0,#6,FILTER13_2
                  INC R0
                  MOV A,@R0
                  SUBB A,#4
                  JNC FILTER14
                  CJNE @R0,#3,FILTER13_2
                  INC R0
                  MOV A,@R0
                  SUBB A,#9
```

```
                JNC FILTER14
                CJNE @R0,#8,FILTER13_2
                INC R0
                MOV A,@R0
                SUBB A,#4
                JNC FILTER14
          FILTER13_2:
                      SETB DC3
                SETB DC2
                CLR DC1
                      CLR DC0
                      JMP EXIT5
          FILTER14:
                      SETB DC3
                SETB DC2
                CLR DC1
                      SETB DC0
          EXIT5:
                RET
```

13.7 Chương trình tổng hợp: (Minh)

```
ORG 00H
LCD_rs      BIT     P2.5
LCD_rw      BIT     P2.6
LCD_en      BIT     P2.7
LCD_D7          BIT     P0.7
LCD_data    EQU  P0
SAVE_COUNT  EQU     50H
BIT11       BIT     51H                  ; KIEM TRA PHAT XUNG SIN
HAY XUNG VUONG BIT11=0 LA PHAT VUONG,BIT11=1 LA PHAT SIN
DC3             EQU        P3.5
DC2             EQU        P3.4
DC1             EQU        P3.3
DC0             EQU        P3.2
EN              EQU        P3.6

MAIN:
      CLR  BIT11
      CALL STARTCOM
      CALL LCD
      CALL CONTROL
      SJMP $

PC_LCD:
           LCALL      LCD_clear
           MOV        DPTR,#Default
           LCALL      LCD_sendstring
           MOV        R0,#30H
           MOV        R1,#7
DisplayNumber:
           MOV        A,@R0
           ADD        A,#48
           LCALL      LCD_senddata
           LCALL      DELAY
           INC        R0
           DJNZ       R1, DisplayNumber
     RET
```

```
;============CHUONG TRINH BAT DAU TU DAY, KIEM TRA BAN PHIM TRUOC
TIEN============================
LCD:
           LCALL      LCD_init
            LCALL     LCD_title
     RET
;=================KHOI_TAO==============
LCD_init:
        MOV     A,#38H          ;Function set: 2 Line, 8-bit, 5x7
dots
        LCALL   LCD_command
        MOV     A,#0EH          ;Display on, Cursor command
        LCALL   LCD_command
        MOV     A,#01H     ;Clear LCD
        LCALL   LCD_command
           RET
;=========HIEN THI TEN NHOM VA DE TAI=============
LCD_title:
           MOV        DPTR,#Title
           MOV        R1,#11
back:      LCALL      LCD_sendstring
           LCALL      LCD_clear
           INC        DPTR
           DJNZ       R1,back
     RET

;=========KIEM TRA DIEU KHIEN BANG PC HAY KEYBOARD===========

;==========QUET PHIM, KHOI DIEU KHIEN TRUNG
TAM==============================
CONTROL:
scan_key1:
        LCALL   IN_HEX
        JB      10,KEY_pc1  ; quet phim den khi co phim an
        SJMP    scan_key1   ; 10 DC SET THI SE GOI CHUONG TRINH
KEY_pc1, KO QUAY LAI KIEM TRA PHIM DAU TIEN AN TIEP
;================================================
KEY_pc1:
           CJNE  R6,#14,KEY_keyboard1
           CALL  PC_control       ; Neu an nut 14 thi goi PC_control
KEY_keyboard1:
           CJNE  R6,#15,scan_key1
           CALL  KEYBOARD         ; Neu an nut 15 thi goi KEYBOARD
     RET

;=============FROM KEYBOARD TO LCD=======
KEYBOARD:
;==========CREATE RAM====================
        MOV     R0,#30H
        MOV     R1,#10
DEL:
           MOV     @R0,#0         ; nap 0 vao vi tri duoc tro boi R0
        INC     R0
        DJNZ    R1,DEL
;=======Display f(Hz)= ==================
           MOV        DPTR,#Default
           LCALL      LCD_sendstring
           MOV        A,#86H                                  ; Place
the cursor at the 7th position of the first line
           LCALL      LCD_command
```

```
           MOV        R0,#30H
           MOV        R1,#0
;=========================================
scan_key:
        LCALL   IN_HEX
        JB      10,KEY_clear
        JMP     scan_key
;=======Press function buttons==============
KEY_clear:
        CJNE    R6,#10,KEY_start            ; PHIM 10 LA PHIM CLEAR
        LCALL   LCD_clear
        CALL    KEYBOARD                            ; GOI LAI
KEYBOARD DE CHAY LAI TU DAU KHAU NHAP GIA TRI TU BAN PHIM
KEY_start:
           CJNE       R6,#11,KEY_wave                    ; PHIM 11 LA
PHIM START
           JB         BIT11,GuiDuLieu
           MOV        DPTR,#square
           CALL       cursor
           CALL       DELAY
           CALL       GuiDuLieu
GuiDuLieu:
           LCALL   LCD_clear
; Display 'Data sent. Press TT to cont'
           MOV        DPTR,#sent
           LCALL LCD_sendstring
           MOV     A,#0C0H ; Place the cursor at the 1st position
of the SECOND line
           LCALL      LCD_command
           MOV        DPTR,#Continue
           LCALL      LCD_sendstring
           CALL    DATA_SENT
           CALL    INIT_UNIQUE1
           CALL    DANGSONG
           CALL    CheckTT
; Check whether TT is pressed
CheckTT:
        LCALL   IN_HEX
        JB      10,KEY_cont1
        JMP     CheckTT

KEY_cont1:
           CJNE       R6,#13,CheckTT
           LCALL      LCD_clear
           JMP        MAIN
      RET
KEY_wave:
           CJNE       R6,#12,KEY_cont2      ; PHIM KIEM TRA DANG
SONG LA PHIM 12
           SETB       BIT11                              ; PHIM 12 DC
AN LA PHAT DANG SONG SIN NEN SETB BIT11=1
           MOV        DPTR,#sin
           ACALL      cursor
           JMP        scan_key

KEY_cont2:
           CJNE       R6,#13,KEY_pc2              ; KIEM TRA PHIM
13 LA PHIM TIEP TUC CO DC AN KO
           CALL       LCD_clear
           JMP        MAIN
```

```
     //    JMP        scan_key

KEY_pc2:
           CJNE       R6,#14,KEY_keyboard2
           JMP        scan_key

KEY_keyboard2:
           CJNE       R6,#15,CONT
           JMP        scan_key

cursor:
           MOV        A,#0C0H                              ; Place
the cursor at the 1st position of the SECOND line
           LCALL      LCD_command
           LCALL      LCD_sendstring
           MOV        A,R1
           ADD        A,#86H
           LCALL      LCD_command
     RET

;=======Press number buttons==============
CONT:      MOV     A,R6
           MOV     R5,A
rescan_key:
           LCALL   IN_HEX
           JB      10,rescan_key
           MOV     A,R5
;=============Save&Display Number==========
        MOV     @R0,A
           ADD        A,#48
        LCALL   LCD_senddata
        INC     R0
        INC     R1
        JMP    scan_key
     RET

;=========CAC CHUONG TRINH CON=============
;============BUSY_FLAG====================
LCD_busy:
         SETB   LCD_D7        ;Make D7th bit of LCD data port as
i/p
           LR    LCD_rs        ;RS=0, Select command register
           SETB   LCD_rw        ;RW=1, we are reading
check:
           CLR    LCD_en        ;Enable H->L
           SETB   LCD_en
           JB     LCD_D7,check  ;read busy flag again till it
becomes 0
    RET                        ;Return from busy routine
;=============COMMAND====================
LCD_command:
           LCALL   LCD_busy
           CLR     LCD_rs        ; RS=0, IR selected
           CLR     LCD_rw        ; RW=0, write to LCD
           MOV     LCD_data,A
           SETB    LCD_en
           CLR     LCD_en
    RET
;=============CLEAR LCD====================
LCD_clear:
```

```
          MOV     A,#01H        ;Clear LCD
        LCALL   LCD_command
    RET
;=============HIENTHI=====================
LCD_senddata:
        LCALL   LCD_busy    ;Wait for LCD to process the data
        SETB    LCD_rs           ;RS=1, DR selected
        CLR     LCD_rw           ;RW=0, write to LCD
       MOV     LCD_data,A
       SETB    LCD_en
        CLR     LCD_en
    RET
;=============SEND STRING==================
LCD_sendstring:
         CLR    A                ;clear Accumulator for any
previous data
         MOVC   A,@A+DPTR        ;load the first character in
accumulator
           JZ         EXIT_send         ;go to exit if zero
           LCALL      LCD_senddata      ;send first char
           LCALL      DELAY             ;create delay time
           INC        DPTR              ;increment data pointer
           SJMP       LCD_sendstring    ;jump back to send the next
character
EXIT_send:
    RET
;================DELAY====================
DELAY:
           MOV   R5,#100
L3:        CALL  DELAYMS
           DJNZ  R5,L3
     RET
DELAYMS:                    ;Xtal=24MHz, Tm=0.5us
           MOV     R6,#3
L1:        MOV     R7,#250
L2:        DJNZ    R7,L2
           DJNZ    R6,L1
           RET
;===========SCAN BAN PHIM==================
IN_HEX:
        MOV     R3,#70
BACK1:  LCALL   GET_KEY
        JNB     10,EXP1
        DJNZ    R3,BACK1
        MOV     R3,#50
BACK2:  LCALL   GET_KEY
        DJNZ    R3,BACK2
EXP1:   NOP
    RET
GET_KEY:
        MOV     A,#0FEH
        MOV     R2,#0
SCAN_ROW:
        MOV     P1,A
        MOV     R4,A
        JNB     P1.4,ROW_0
        JNB     P1.5,ROW_1
        JNB     P1.6,ROW_2
        JNB     P1.7,ROW_3
        MOV     A,R4
```

```
        RL      A
        INC     R2
        CJNE    R2,#4,SCAN_ROW
        SJMP    NO_CODE
ROW_0:  MOV     A,R2
        ADD     A,#0
        SETB    10
        MOV     R6,A
        SJMP    EXIT
ROW_1:  MOV     A,R2
        ADD     A,#4
        SETB    10
        MOV     R6,A
        SJMP    EXIT
 ROW_2: MOV     A,R2
        ADD     A,#8
        SETB    10
        MOV     R6,A
        SJMP    EXIT
 ROW_3: MOV     A,R2
        ADD     A,#12
        SETB    10
        MOV     R6,A
        SJMP    EXIT
NO_CODE:
        CLR 10
EXIT:
    RET
;==============TRA_BANG===============
 Title:
 DB        'HELLO',00H
 DB   'TO 1 - NHOM 9',00H
 DB   'NG THI BAO TRAM',00H
 DB   'NG LE HOANG TUAN',00H
 DB   'VO NGOC TRI MINH',00H
 DB   'TRAN PHUOC THINH',00H
 DB   'TRUONG KIM NGA',00H
 DB   'SENGSAY SOMSAMAY',00H
 DB   'MACH PHAT TAN SO',00H
 DB   'DOI TUONG DKHIEN',00H
 DB   '14.PC OR 15.KEY',00H
 Default:
 DB   'f(Hz)=',00H
 sin:
 DB   'waveform: sine',00H
 square:
 DB 'waveform: square',00H
 sent:
 DB 'Data sent',00H
 Continue:
 DB 'Press TT to continue',00H

 /*************************CHUONG TRINH NHAN DU LIEU TU TREN MAY
TINH GUI XUONG VDK DE HIEN
THI*************************************/

 PC_control:
    CALL NHAN_DULIEU
      CALL PC_LCD
      CALL CHUYENDOI
```

```
      MOV R0,#30H
//    CALL INIT_UNIQUE1
      CALL DANGSONG
  RET
CHUYENDOI:
    MOV R0,#30H                    ; VUNG NHO CU
//    MOV SAVE_COUNT,R1            ; BIEN SAVE_COUNT LUU BO DEM SO
LUONG SO NHAP VAO SAVE_COUNT=R1
      CLR C                        ; XOA CO C
      MOV A,#7
      SUBB A,SAVE_COUNT            ; A=A-R1 VOI R1 LUU BIEN DEM SO
LUONG CAC SO NHAP VAO
      MOV R5,A                     ; R5=7-SAVE_COUNT
      MOV R1,#40H                       ; VUNG NHO MOI
STEP1:
      INC R0
      DJNZ R5,STEP1
      MOV R6,SAVE_COUNT
STEP2:
    MOV A,@R0
      MOV @R1,A
      INC R0
      INC R1
      DJNZ R6,STEP2
      MOV R1,#40H
      MOV R0,#30H
      MOV R6,SAVE_COUNT
STEP3:
    MOV A,@R1
      MOV @R0,A
      INC R0
      INC R1
      DJNZ R6,STEP3
  RET

NHAN_DULIEU:
    CALL WAVEFORM_GT         ; KIEM TRA DANG SONG GUI XUONG
      CJNE R0,#49H,VUONG_GT  ; R0=55H LA PHAT SONG VUONG
      CALL SIN_GT                       ; R0=49H LA PHAT SONG SIN
  RET
WAVEFORM_GT:                       ; LAY GIA TRI KIEU DANG SONG GUI
XUONG VA SO LUONG KI TU GUI XUONG LUU VAO SAVE_COUNT
      JNB RI,$                     ; DOI KIEU DANG SONG GUI XUONG
      MOV A,SBUF
      CLR RI
      MOV R0,A

      JNB RI,$                     ; DOI SO LUONG KI TU GUI XUONG LUU
VAO SAVE_COUNT
      MOV A,SBUF
      CLR RI
      MOV R1,A
      MOV SAVE_COUNT,R1
  RET
VUONG_GT:
      CLR  BIT11                   ; NEU LA SONG VUONG THI CLEAR
BIT11=0
      CALL INIT_UNIQUE1            ; KHOI TAO VUNG LUU GIA TRI DU LIEU
GUI XUONG
      CALL CHO_NHAN
```

```
  RET
SIN_GT:
    SETB BIT11                    ; NEU LA SONG SIN THI SETB BIT11=1
      CALL INIT_UNIQUE1           ; KHOI TAO VUNG LUU GIA TRI DU LIEU
GUI XUONG
      CALL CHO_NHAN
  RET

/**************LUU DU LIEU TAN SO CAN PHAT TU PC XUONG VDK TAI VUNG
NHO TU 30H DEN 36H***********/
CHO_NHAN:                   ; CHO` NHAN DU LIEU GUI XUONG TIEP TUC
TU PC
    JNB RI,$
      MOV A,SBUF
      CLR RI
      MOV @R0,A
      INC R0
      DJNZ R5,CHO_NHAN
  RET
/*SAU KHUC TREN NAY CHO LCD HIEN THI VA PHAT XUNG CHI VIEC LAY CAC
GIA TRI LUU TRONG VUNG NHO RAM NAY RA DE HIEN THI VA TINH TOAN*/

/******************CHUONG TRINH PHAT
XUNG*************************************************************
**************************/
DANGSONG:
              CLR EN
              JB BIT11,PHATSIN
PHATVUONG:
           SETB DC3
           SETB DC2
           SETB DC1
           CLR  DC0
           CALL PHATXUNG
           JMP  EXITMAIN
PHATSIN:
           CALL CHONBANG
           CALL PHATXUNG
           JMP EXITMAIN
CHONBANG:
           MOV R1,SAVE_COUNT
           CJNE R1,#2,CHON1
           CALL XULY1
           JMP EXIT_CB
      CHON1:
           CJNE R1,#3,CHON2
           CALL XULY2
           JMP EXIT_CB
      CHON2:
           CJNE R1,#4,CHON3
           CALL XULY3
           JMP EXIT_CB
      CHON3:
           CJNE R1,#5,CHON4
           CALL XULY4
           JMP EXIT_CB
      CHON4:
           CALL XULY5
      EXIT_CB:
           RET
```

```
XULY1:
            MOV R0,#30H
            MOV A,@R0
            MOV B,#10
            MUL AB
            INC R0
            ADD A,@R0
            MOV R2,A
            SUBB A,#40
            JNC XL1_1
      FILTER1:
            CLR DC3
            CLR DC2
            CLR DC1
            CLR DC0
            JMP EXIT1
      XL1_1:
            MOV A,R2
            SUBB A,#80
            JNC FILTER3_1
      FILTER2:
            CLR DC3
            CLR DC2
            CLR DC1
            SETB DC0
            JMP EXIT1
      FILTER3_1:
            CLR DC3
            CLR DC2
            SETB DC1
            CLR DC0
      EXIT1:
            RET
XULY2:
            MOV R0,#30H
            MOV A,@R0
            SUBB A,#2
            JNC XL2_1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC FILTER4
      FILTER3_2:
            CLR DC3
            CLR DC2
            SETB DC1
            CLR DC0
            JMP EXIT2
      XL2_1:
            MOV R0,#30H
            MOV A,@R0
            SUBB A,#4
            JNC XL2_2
            CJNE @R0,#3,FILTER4
            INC R0
            MOV A,@R0
            SUBB A,#2
            JNC  FILTER5
      FILTER4:
            CLR DC3
```

```
            CLR DC2
            SETB DC1
            SETB DC0
            JMP EXIT2
      XL2_2:
            MOV R0,#30H
            MOV A,@R0
            SUBB A,#7
            JNC FILTER6_1
            CJNE @R0,#6,FILTER5
            INC R0
            MOV A,@R0
            SUBB A,#4
            JNC FILTER6_1
      FILTER5:
            CLR DC3
            SETB DC2
            CLR DC1
            CLR DC0
            JMP EXIT2
      FILTER6_1:
            CLR DC3
            SETB DC2
            CLR DC1
            SETB DC0

      EXIT2:
            RET


XULY3:
                  MOV R0,#30H
                  MOV A,@R0
                  SUBB A,#2
                        JNC XL3_1
                  INC R0
                  MOV A,@R0
                  SUBB A,#3
                  JNC FILTER7
                  CJNE @R0,#2,FILTER6_2
                  INC R0
                  MOV A,@R0
                  SUBB A,#8
                  JNC FILTER7
            FILTER6_2:
                  CLR DC3
                  SETB DC2
                  CLR DC1
                        SETB DC0
                        JMP EXIT3

XL3_1:
                  MOV R0,#30H
                  MOV A,@R0
                  SUBB A,#3
                        JNC XL3_2
                  CJNE @R0,#2,FILTER7
                  INC R0
                  MOV A,@R0
                  SUBB A,#6
```

```
                JNC FILTER8
                CJNE @R0,#5,FILTER7
                INC R0
                MOV A,@R0
                SUBB A,#6
                JNC FILTER8
          FILTER7:
                CLR DC3
                SETB DC2
                SETB DC1
                      CLR DC0
                      JMP EXIT3
XL3_2:
                MOV R0,#30H
                MOV A,@R0
                SUBB A,#6
                      JNC FILTER9_1
                CJNE @R0,#5,FILTER8
                INC R0
                MOV A,@R0
                SUBB A,#2
                JNC FILTER9_1
                CJNE @R0,#1,FILTER8
                INC R0
                MOV A,@R0
                SUBB A,#2
                JNC FILTER9_1
          FILTER8:
                CLR DC3
                SETB DC2
                SETB DC1
                      SETB DC0
                      JMP EXIT3
          FILTER9_1:
                SETB DC3
                CLR DC2
                CLR DC1
                      CLR DC0
          EXIT3:
                RET
XULY4:
                MOV R0,#30H
                MOV A,@R0
                SUBB A,#2
                JNC XL4_1
                INC R0
                MOV A,@R0
                SUBB A,#1
                JNC FILTER10
                INC R0
                MOV A,@R0
                SUBB A,#3
                JNC FILTER10
                CJNE @R0,#2,FILTER9_2
                INC R0
                MOV A,@R0
                SUBB A,#4
                JNC FILTER10
          FILTER9_2:
                      SETB DC3
```

```
                CLR DC2
                CLR DC1
                      CLR DC0
                      JMP EXIT4

XL4_1:
                MOV R0,#30H
                MOV A,@R0
                SUBB A,#3
                JNC XL4_2
                CJNE @R0,#2,FILTER10
                INC R0
                MOV A,@R0
                SUBB A,#1
                JNC FILTER11
                INC R0
                MOV A,@R0
                SUBB A,#5
                JNC FILTER11
                CJNE @R0,#4,FILTER10
                INC R0
                MOV A,@R0
                SUBB A,#8
                JNC FILTER11
          FILTER10:
                      SETB DC3
                CLR DC2
                CLR DC1
                      SETB DC0
                      JMP EXIT4
XL4_2:
                MOV R0,#30H
                MOV A,@R0
                SUBB A,#5
                JNC XL4_3
                CJNE @R0,#4,FILTER11
                INC R0
                MOV A,@R0
                SUBB A,#1
                JNC FILTER12
                INC R0
                MOV A,@R0
                SUBB A,#10
                JNC FILTER12
                CJNE @R0,#9,FILTER11
                INC R0
                MOV A,@R0
                SUBB A,#6
                JNC FILTER12
          FILTER11:
                      SETB DC3
                CLR DC2
                SETB DC1
                      CLR DC0
                      JMP EXIT4
XL4_3:
                MOV R0,#30H
                MOV A,@R0
                SUBB A,#9
                JNC FILTER13_1
```

```
                CJNE @R0,#8,FILTER12
                INC R0
                MOV A,@R0
                SUBB A,#2
                JNC FILTER13_1
                CJNE @R0,#1,FILTER12
                INC R0
                MOV A,@R0
                SUBB A,#10
                JNC FILTER13_1
                CJNE @R0,#9,FILTER12
                INC R0
                MOV A,@R0
                SUBB A,#2
                JNC FILTER13_1
          FILTER12:
                SETB DC3
                CLR DC2
                SETB DC1
                      SETB DC0
                      JMP EXIT4
          FILTER13_1:
                      SETB DC3
                SETB DC2
                CLR DC1
                      CLR DC0
          EXIT4:
                RET
XULY5:
                MOV R0,#30H
                MOV A,@R0
                SUBB A,#2
                JNC FILTER14
                INC R0
                MOV A,@R0
                SUBB A,#7
                JNC FILTER14
                CJNE @R0,#6,FILTER13_2
                INC R0
                MOV A,@R0
                SUBB A,#4
                JNC FILTER14
                CJNE @R0,#3,FILTER13_2
                INC R0
                MOV A,@R0
                SUBB A,#9
                JNC FILTER14
                CJNE @R0,#8,FILTER13_2
                INC R0
                MOV A,@R0
                SUBB A,#4
                JNC FILTER14
          FILTER13_2:
                      SETB DC3
                SETB DC2
                CLR DC1
                      CLR DC0
                      JMP EXIT5
          FILTER14:
                      SETB DC3
```

```
                SETB DC2
                CLR DC1
                      SETB DC0
          EXIT5:
                RET

     PHATXUNG:
         MOV R1,SAVE_COUNT
           CJNE R1,#2,PHAT1
           CALL TSCHUC
           JMP EXIT_PX
     PHAT1:
           CJNE R1,#3,PHAT2
           CALL TSTRAM
           JMP EXIT_PX
     PHAT2:
           CJNE R1,#4,PHAT3
           CALL TSNGAN
           JMP EXIT_PX
     PHAT3:
           CJNE R1,#5,PHAT4
           CALL TSCHUCNGAN
           JMP EXIT_PX
     PHAT4:
           CALL TSTRAMNGAN
     EXIT_PX:
           RET


TSCHUC:
           CALL TINH3
     CHUC1:
           CALL TINH1
           MOV R6,A
           SUBB A,#13H
           JC CHUC1
           CJNE R6,#13H,CHUC2
           MOV A,R1
           SUBB A,#89H
           JC CHUC1
     CHUC2:
           DEC R4
           MOV A,R4
     CHUC3:
           MOV R5,A
     CHUC4:
           CALL DELAY99US
           DJNZ R5,CHUC4
           CPL P3.7
           SJMP  CHUC3
           RET
TSTRAM:
           CLR C
           MOV R0,#30H
           MOV A,@R0
           SUBB A,#3
           JNC TRAM6
           CJNE @R0,#2,TRAM1
           INC R0
           MOV A,@R0
```

```
            SUBB A,#6
            JNC TRAM6
            CJNE @R0,#5,TRAM1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC TRAM6
      TRAM1:
            CALL TINH4
      TRAM2:
            CALL TINH1
            MOV R6,A
            SUBB A,#61H
            JC TRAM2
            CJNE R6,#61H,TRAM3
            MOV A,R1
            SUBB A,#0A9H
            JC TRAM2
      TRAM3:
            DEC R4
            MOV A,R4
      TRAM4:
            MOV R5,A
      TRAM5:
            CALL DELAY19US
            DJNZ R5,TRAM5
            CPL P3.7
            SJMP TRAM4
            JMP EXIT_TRAM
      TRAM6:
            CALL TINH3
      TRAM7:
            CALL TINH1
            MOV R6,A
            SUBB A,#18H
            JC TRAM7
            CJNE R6,#18H,TRAM8
            MOV A,R1
            SUBB A,#06BH
            JC TRAM7
      TRAM8:
            DEC R4
            MOV A,R4

      TRAM9:
            MOV R5,A
      TRAM10:
            CALL DELAY7US
            DJNZ R5,TRAM10
            CPL P3.7
            SJMP TRAM9
      EXIT_TRAM:
            RET
TSNGAN:
            CLR C
            MOV R0,#30H
            MOV A,@R0
            SUBB A,#3
            JNC NGAN6
            CJNE @R0,#2,NGAN1
```

```
      INC R0
      MOV A,@R0
      SUBB A,#6
      JNC NGAN6
      CJNE @R0,#5,NGAN1
      INC R0
      MOV A,@R0
      SUBB A,#6
      JNC NGAN6
NGAN1:
      CALL TINH4
NGAN2:
      CALL TINH1
      MOV R6,A
      SUBB A,#61H
      JC NGAN2
      CJNE R6,#61H,NGAN3
      MOV A,R1
      SUBB A,#0A9H
      JC NGAN2
NGAN3:
      DEC R4
      MOV A,R4
NGAN4:
      MOV R5,A
NGAN5:
      NOP
      NOP
      DJNZ R5,NGAN5
      CPL P3.7
      SJMP NGAN4
      JMP EXIT_NGAN
NGAN6:
      CALL TINH3
      SUBB A,#40
      JC NGAN13
NGAN7:
      CALL TINH1
      MOV R6,A
      SUBB A,#27H
      JC NGAN7
      CJNE R6,#27H,NGAN8
      MOV A,R1
      SUBB A,#11H
      JC NGAN7
NGAN8:
      MOV A,R4
      SUBB A,#5
      MOV R4,A
      CALL TINH2
      CJNE A,#0,NGAN11
      MOV A,R4
NGAN9:
      MOV R5,A
NGAN10:
      DJNZ R5,NGAN10
      CPL P3.7
      SJMP NGAN9
      JMP EXIT_NGAN
NGAN11:
```

```
            MOV A,R4
            MOV R5,A
      NGAN12:
            DJNZ R5,NGAN12
            CPL P3.7
            SJMP NGAN11
            JMP EXIT_NGAN
      NGAN13:
            CALL TINH1
            MOV R6,A
            SUBB A,#13H
            JC NGAN13
            CJNE R6,#13H,NGAN14
            MOV A,R1
            SUBB A,#89H
            JC NGAN13
      NGAN14:
            DEC R4
            MOV A,R4
      NGAN15:
            MOV R5,A
      NGAN16:
            DJNZ R5,NGAN16
            CPL P3.7
            SJMP NGAN15
      EXIT_NGAN:
            RET

TSCHUCNGAN:
            CLR C
            MOV R0,#30H
            MOV A,@R0
            SUBB A,#3
            JNC CNGAN8
            CJNE @R0,#2,CNGAN1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC CNGAN8
            CJNE @R0,#5,CNGAN1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC CNGAN8
      CNGAN1:
            CALL TINH4
      CNGAN2:
            CALL TINH1
            MOV R6,A
            SUBB A,#27H
            JC CNGAN2
            CJNE R6,#27H,CNGAN3
            MOV A,R1
            SUBB A,#11H
            JC CNGAN2
      CNGAN3:
            MOV A,R4
            SUBB A,#5
            MOV R4,A
            CALL TINH2
```

```
            CJNE A,#0,CNGAN6
            MOV A,R4
      CNGAN4:
            MOV R5,A
      CNGAN5:
            DJNZ R5,CNGAN5
            CPL P3.7
            SJMP CNGAN4
            JMP EXIT_CNGAN
      CNGAN6:
            MOV A,R4
            MOV R5,A
      CNGAN7:
            DJNZ R5,CNGAN7
            CPL P3.7
            SJMP CNGAN6
            JMP EXIT_CNGAN
      CNGAN8:
            CALL TINH3
      CNGAN9:
            CALL TINH1
            MOV R6,A
            SUBB A,#03H
            JC CNGAN9
            CJNE R6,#03H,CNGAN3
            MOV A,R1
            SUBB A,#0E9H
            JC CNGAN9
            JMP CNGAN3
      EXIT_CNGAN:
            RET
TSTRAMNGAN:
            CLR C
            MOV R0,#30H
            MOV A,@R0
            SUBB A,#3
            JNC TNGAN12
            CJNE @R0,#2,TNGAN1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC TNGAN12
            CJNE @R0,#5,TNGAN1
            INC R0
            MOV A,@R0
            SUBB A,#6
            JNC TNGAN12
      TNGAN1:
            CALL TINH4
      TNGAN2:
            CALL TINH1
            MOV R6,A
            SUBB A,#03H
            JC TNGAN2
            CJNE R6,#03H,TNGAN3
            MOV A,R1
            SUBB A,#0E9H
            JC TNGAN2
      TNGAN3:
            MOV A,R4
```

```
            SUBB A,#5
            MOV R4,A
            CJNE R4,#0,TNGAN5
      TNGAN4:
            NOP
            CPL P3.7
            SJMP TNGAN4
            JMP EXIT_TNGAN
      TNGAN5:
            CJNE R4,#1,TNGAN7
      TNGAN6:
            NOP
            NOP
            CPL P3.7
            SJMP TNGAN6
            JMP EXIT_TNGAN
      TNGAN7:
            CALL TINH2
            CJNE A,#0,TNGAN10
            MOV A,R4
      TNGAN8:
            MOV R5,A
      TNGAN9:
            DJNZ R5,TNGAN9
            CPL P3.7
            SJMP TNGAN8
            JMP EXIT_TNGAN
      TNGAN10:
            MOV A,R4
            MOV R5,A
      TNGAN11:
            DJNZ R5,TNGAN11
            CPL P3.7
            SJMP TNGAN10
            JMP EXIT_TNGAN
      TNGAN12:
            CALL TINH3
            SUBB A,#26
            JC TNGAN14
      TNGAN13:
            CPL P3.7
            SJMP TNGAN13
            JMP EXIT_TNGAN
      TNGAN14:
            NOP
            CPL P3.7
            SJMP TNGAN14
      EXIT_TNGAN:
            RET
TINH1:
            CLR C
            INC R4
            MOV B,R4
            MOV A,R3
            MUL AB
            MOV R1,A
            MOV A,B
            RET
TINH2:
            MOV A,R4
```

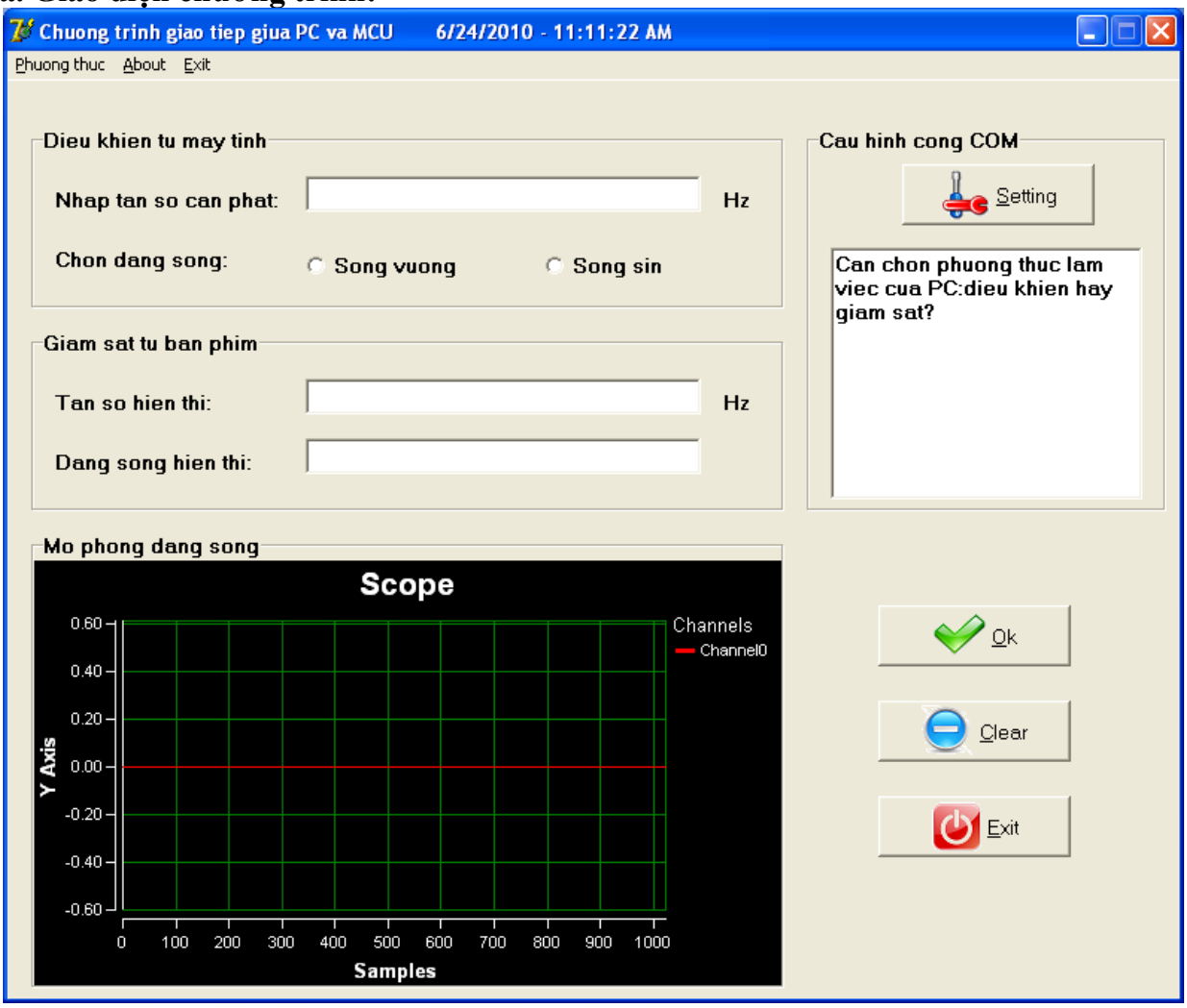
```
            MOV B,#2
            DIV AB
            MOV R4,A
            MOV A,B
            RET
TINH3:
            MOV R0,#30H
            MOV A,@R0
            MOV B,#10
            MUL  AB
            INC R0
            ADD A,@R0
            MOV R3,A
            MOV R4,#0
            RET


TINH4:
            MOV R0,#30H
            MOV A,@R0
            MOV B,#10
            MUL AB
            INC R0
            ADD A,@R0
            MOV B,#10
            MUL AB
            INC R0
            ADD A,@R0
            MOV R3,A
            MOV R4,#0
            RET
DELAY99US:
            NOP
            MOV R6,#96
      DL99:
            DJNZ  R6,DL99
            RET
DELAY19US:
            NOP
            MOV R6,#16
      DL19:
            DJNZ R6,DL19
            RET
DELAY7US:
            NOP
            MOV R6,#4
      DL7:
            DJNZ R6,DL7
            RET
EXITMAIN:
  RET
/******************CHUONG TRINH XUAT DU LIEU TU KEYBOARD LEN
PC*************************************************************
****/
DATA_SENT:
  CALL STARTCOM
  CJNE R1,#7,TAORAM1
  CALL TAORAM2
 RET
STARTCOM:
```

```
    MOV SCON,#52H     ; KHOI DONG PORT NOI TIEP CHE DO 1
      MOV TMOD,#20H   ; KHOI DONG BO DINH THOI 1-CHE DO 2
      MOV TH1,#(-13)  ; GIA TRI NAP LAI DE CO 4800 BAUD VOI THACH
ANH 24MHz
      SETB TR1        ; BO DINH THOI 1 HOAT DONG
  RET
TAORAM2:
    CALL INIT_UNIQUE1
      CALL XUAT
  RET
TAORAM1:
    MOV R0,#30H       ; VUNG NHO CU
      MOV SAVE_COUNT,R1  ; BIEN SAVE_COUNT LUU BO DEM SO LUONG SO
NHAP VAO SAVE_COUNT=R1
      CLR C           ; XOA CO C
      MOV A,#7
      SUBB A,SAVE_COUNT          ; A=A-R1 VOI R1 LUU BIEN DEM SO
LUONG CAC SO NHAP VAO
      MOV R5,A        ; R5=7-SAVE_COUNT
      MOV R1,#40H           ; VUNG NHO MOI
INIT_ZERO:
      MOV @R1,#0
      INC R1
    DJNZ R5,INIT_ZERO
INIT_DATA:
      MOV A,@R0
      MOV @R1,A
      INC R1
      INC R0
      DJNZ SAVE_COUNT,INIT_DATA
      CALL INIT_UNIQUE
      CALL XUAT
  RET
INIT_UNIQUE1:
      MOV R0,#30H
      MOV R5,#7
  RET
INIT_UNIQUE:
    MOV R0,#40H
      MOV R5,#7
  RET
XUAT:
    JNB TI,$
      CLR TI
      MOV SBUF,#61H
      CALL DATA_UP
      JNB TI,$
      CLR TI
      MOV SBUF,#7AH
      RET
DATA_UP:
   JNB TI,$
   CLR TI
   MOV A,@R0
   MOV SBUF,A
   INC R0
   DJNZ R5,DATA_UP
   RET
END
```

13.8 Chương trình Delphi: (Minh)

a. Giao diện chương trình:



b. Code Delphi:

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  Dialogs, Menus, StdCtrls, Buttons, ComCtrls, LPComponent,
  SLComponentCollection, LPDrawLayers, SLScope, SLCommonGen, SLSignalGen,
  CPort;
type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    Help1: TMenuItem;
    About1: TMenuItem;
    Exit1: TMenuItem;
    GroupBox1: TGroupBox;
    Label1: TLabel;
    Edit1: TEdit;
    Label2: TLabel;
    Label3: TLabel;
    RadioButton2: TRadioButton;
    GroupBox2: TGroupBox;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
```

```
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    GroupBox3: TGroupBox;
    BitBtn4: TBitBtn;
    RadioButton3: TRadioButton;
    GroupBox4: TGroupBox;
    SLSignalGen1: TSLSignalGen;
    SLScope1: TSLScope;
    ComDataPacket1: TComDataPacket;
    ComPort1: TComPort;
    Memo1: TMemo;
    Edit2: TEdit;
    Edit3: TEdit;
    Maytinh1: TMenuItem;
    Banphim1: TMenuItem;
    procedure Exit1Click(Sender: TObject);
    procedure About1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
    procedure RadioButton1Click(Sender: TObject);
    procedure RadioButton2Click(Sender: TObject);
    procedure RadioButton3Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure Maytinh1Click(Sender: TObject);
    procedure Banphim1Click(Sender: TObject);
    procedure ComDataPacket1Packet(Sender: TObject; const Str: String);

  private
    { Private declarations }
  public
    { Public declarations }
  end;
  const
    startbyte=$61; // Khai bao tin hieu nhan hand shaking bat dau la 'a'
    stopbyte=$7A;  // Khai bao tin hieu nhan hand shaking ket thuc la 'z'

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.Exit1Click(Sender: TObject);
begin
  ComPort1.Close ;
  Close;
end;

procedure TForm1.About1Click(Sender: TObject);
begin
messagedlg('Chuong trinh giao tiep giua PC - VDK qua cong COM'+chr(13)+
'Nhiem vu: Dieu khien va giam sat mach phat tan so 20Hz-1MHz'+chr(13)+
'Nhom 01 - Lop 06DT1'+chr(13)+
'Gom cac thanh vien:'+chr(13)+chr(13)+
'01. Vo Ngoc Tri Minh'+chr(13)+
'02. Truong Thi Kim Nga'+chr(13)+
'03. Tran Phuoc Thinh'+chr(13)+
'04. Nguyen Thi Bao Tram'+chr(13)+
'05. Nguyen Le Hoang Tuan'+chr(13)+
'06. Sengsay Somsamay'+chr(13),mtInformation,[mbOk],0);
end;
```

```
procedure TForm1.BitBtn2Click(Sender: TObject);
begin
Edit1.Text :='';
Edit2.Text :='';
Edit3.Text :='';
Edit1.SetFocus ;
SLSignalGen1.Amplitude :=0;
Memo1.Text :='';
ComPort1.Close ;
ComPort1.Open ;

end;

procedure TForm1.BitBtn4Click(Sender: TObject);
begin
 Comport1.ShowSetupDialog;
 if messagedlg('Chon Ok de luu thong so moi'+chr(13)+'Chon Cancel de lay
lai thong so cu.',mtConfirmation,[mbOk,mbCancel],0)=mrOk then
  begin
   if ComPort1.Connected then
      begin
         messagedlg('Cong COM dang mo'+chr(13)+'Nhan OK de dong cong COM
va mo lai cong COM',mtwarning,[mbOk],0);
         ComPort1.Close;
         ComPort1.Open ;
      end
    else
         ComPort1.Open;
  end;
  ComPort1.Open;
  messagedlg('Cong COM da duoc mo va thiet lap theo thong so ban
chon.',mtwarning,[mbOK],0);
end;

procedure TForm1.BitBtn1Click(Sender: TObject);
var st,st1,st2 :string;
    j,k,tmp,dem:integer;
begin

   st:=Edit1.Text ;
   st2:=Edit1.Text ;
   if (st='') then
     begin
        messagedlg('ban can nhap gia tri tan so can phat
vao',mtError,[mbOk],0);
        Edit1.SetFocus ;
     end
   else
      if (StrToInt(st)<20) or (StrToInt(st)>1000000) then
          begin
              messagedlg('ban can nhap gia tri tan so can phat trong
khoang 20Hz-1MHz',mtError,[mbOk],0);
              Edit1.SetFocus ;
          end
      else
         begin

          if RadioButton2.Checked  =true then
                begin
                  ComPort1.WriteStr(chr($55)); // Xuat tin hieu bao dang
song ra la vuong(U) xuong VDK
                  SLSignalGen1.Start ;
                  SlSignalGen1.Amplitude  :=2;
```

```
                    SLSignalGen1.SignalType :=stsquare;
                    SLSignalGen1.Frequency  :=StrToInt(st2);
                  end
               else
                 begin
                    if RadioButton3.Checked  =true then
                       begin
                        ComPort1.WriteStr(chr($49)); // Xuat tin hieu bao
dang song ra la sin(I) xuong VDK
                        SLSignalGen1.Start ;
                        SlSignalGen1.Amplitude  :=2;
                        SLSignalGen1.SignalType :=sttone;
                        SLSignalGen1.Frequency  :=StrToInt(st2);
                       end
                    else
                        begin
                            messagedlg('Can chon kieu dang song hien
thi',mtError,[mbOk],0);
                            ComPort1.WriteStr(chr($55)); // Xuat tin hieu
bao dang song ra la vuong(U) xuong VDK
                            SLSignalGen1.Start ;
                            SlSignalGen1.Amplitude  :=2;
                            SLSignalGen1.SignalType :=stsquare;
                            SLSignalGen1.Frequency  :=StrToInt(st2);
                        end;
                end;

               begin
                 messagebeep(0);
                 k:=length(Edit1.Text );
                 ComPort1.WriteStr(chr(k));
                 dem:=7-k;    // Bien dem bao' so luong 0 can dien vao

                  if dem=5 then
                    begin
                      ComPort1.WriteStr(chr($0));
                      ComPort1.WriteStr(chr($0));
                      ComPort1.WriteStr(chr($0));
                      ComPort1.WriteStr(chr($0));
                      ComPort1.WriteStr(chr($0));
                    end;
                  if dem=4 then
                    begin
                      ComPort1.WriteStr(chr($0));
                      ComPort1.WriteStr(chr($0));
                      ComPort1.WriteStr(chr($0));
                      ComPort1.WriteStr(chr($0));
                    end;
                  if dem=3 then
                    begin
                      ComPort1.WriteStr(chr($0));
                      ComPort1.WriteStr(chr($0));
                      ComPort1.WriteStr(chr($0));
                    end;
                  if dem=2 then
                    begin
                      ComPort1.WriteStr(chr($0));
                      ComPort1.WriteStr(chr($0));
                    end;
                  if dem=1 then
                    begin
                      ComPort1.WriteStr(chr($0));
                    end;

                 for j:=1 to k do
                  begin
```

```
                    tmp:=StrToInt(st[j]);     // Doi ki tu st[j] sang so
                    st1:=chr(tmp);            // Tim ma ASCII tai vi tri
tmp
                    ComPort1.WriteStr(st1);   // Gui ma ASCII do' xuong VDK
                  end;

                  memo1.Text :='Ban da nhan nut OK. Du lieu da gui xuong
VDK';
            end;
      end;
end;

procedure TForm1.FormActivate(Sender: TObject);
var today:TDatetime;
begin
  ComDataPacket1.StartString :=chr(startbyte);  // Bat dau hand shaking
  ComDataPacket1.StopString  :=chr(stopbyte);   // Ket thuc hand shaking
  ComPort1.BaudRate :=br4800;
  ComPort1.DataBits :=dbEight;
  ComPort1.StopBits :=sbOneStopBit;
  Comport1.Open ;
  today:=now();
  caption:=('Chuong trinh giao tiep giua PC va MCU
')+DateToStr(today)+ ' - '+ TimeToStr(today);
  SLSignalGen1.Amplitude :=0;
  memo1.Text :='Can chon phuong thuc lam viec cua PC:dieu khien hay giam
sat?';
end;

procedure TForm1.RadioButton1Click(Sender: TObject);
begin
  edit1.SetFocus ;
end;

procedure TForm1.RadioButton2Click(Sender: TObject);
begin
edit1.SetFocus ;
end;

procedure TForm1.RadioButton3Click(Sender: TObject);
begin
edit1.SetFocus ;
end;

procedure TForm1.BitBtn3Click(Sender: TObject);
begin
  ComPort1.Close ;
  Close;
end;


procedure TForm1.Maytinh1Click(Sender: TObject);
begin
   ComPort1.Close ;
   ComPort1.Open ;
   ComPort1.WriteStr(chr($50));
   Edit1.Visible :=true;
   Edit1.SetFocus ;
   Bitbtn1.Visible :=true;
   Bitbtn2.Visible :=true;
   memo1.Text :='PC lam nhiem vu dieu khien,nhap tan so can phat vao va
click OK';
end;

procedure TForm1.Banphim1Click(Sender: TObject);
begin
```

```
  ComPort1.Close ;
  ComPort1.Open ;
  ComPort1.WriteStr(chr($4B));
  Edit1.Visible :=false ;
  Bitbtn1.Visible :=false;
  Bitbtn2.Visible :=false;
  memo1.Text :='PC lam nhiem vu giam sat, xin doi tan so phat tu VDK len';
end;

procedure TForm1.ComDataPacket1Packet(Sender: TObject; const Str: String);
var s,str1:string;
    sum:integer;
begin
   str1:=str;

sum:=ord(str1[7])+ord(str1[6])*10+ord(str1[5])*100+ord(str1[4])*1000+ord(s
tr1[3])*10000+ord(str1[2])*100000+ord(str1[1])*1000000;
   s:=IntToStr(sum);
   Edit2.Text :=s;
   Edit3.Text :='GIAO TIEP TU KEYBOARD';
   end;

end.
```