Chương 1: GIỚI THIỆU VI ĐIỀU KHIỂN 8051

1.1. Tổng quan về họ vi điều khiển

Vào năm 1981 hãng Intel giới thiệu bộ vi điều khiển được gọi là 8051. Bộ vi điều khiển này có 128 byte RAM, 4K byte ROM trên chip, hai bộ định thời, một cổng nối tiếp và 4 cổng (độ rộng 8 bit) vào – ra tất cả được đặt trên một chip. 8051 là một bộ xử lý 8 bit có nghĩa là CPU chỉ có thể làm việc với 8 bit dữ liệu tại một thời điểm. Dữ liệu lớn hơn 8 bit được chia ra thành các dự liệu 8 bit để xử lý.

8051 đã trở lên phổ biến sau khi Intel cho phép các nhà sản xuất khác sản xuất và bán các dạng biến thể của 8051. Điều này dẫn đến sự ra đời nhiều phiên bản của 8051 với các tốc độ khác nhau và dung lượng ROM trên chip khác nhau. Mặc dù có nhiều biến thể khác nhau của 8051 về tốc độ và dung lượng nhớ ROM trên chip, nhưng tất cả chúng đều tương thích với 8051 ban đầu về các lệnh. Điều này có nghĩa là nếu viết chương trình của mình cho một phiên bản nào của 8051 thì nó cũng sẽ chạy với mọi phiên bản khác mà không phân biệt nó được sản xuất từ hãng nào.

Đặc Tính	**Số Lượng**
ROM trên chip	4K byte
RAM	128 byte
Bộ định thời	2
Các chân vào – ra	32
Cổng nối tiếp	1
Nguồn ngắt	6

Bảng 1.1: Các đặc tính của 8051 đầu tiên

Ta có thể so sánh 3 loại chip của họ vi điều khiển 51 như bảng sau

Đặc tính	**8051**	**8052**	**8031**
ROM trên chip	4K byte	8K byte	0 K byte
RAM	128 byte	256 byte	128 byte
Bộ định thời	2	3	2
Chân vào - ra	32	32	32
Cổng nối tiếp	1	1	1
Nguồn ngắt	6	8	6

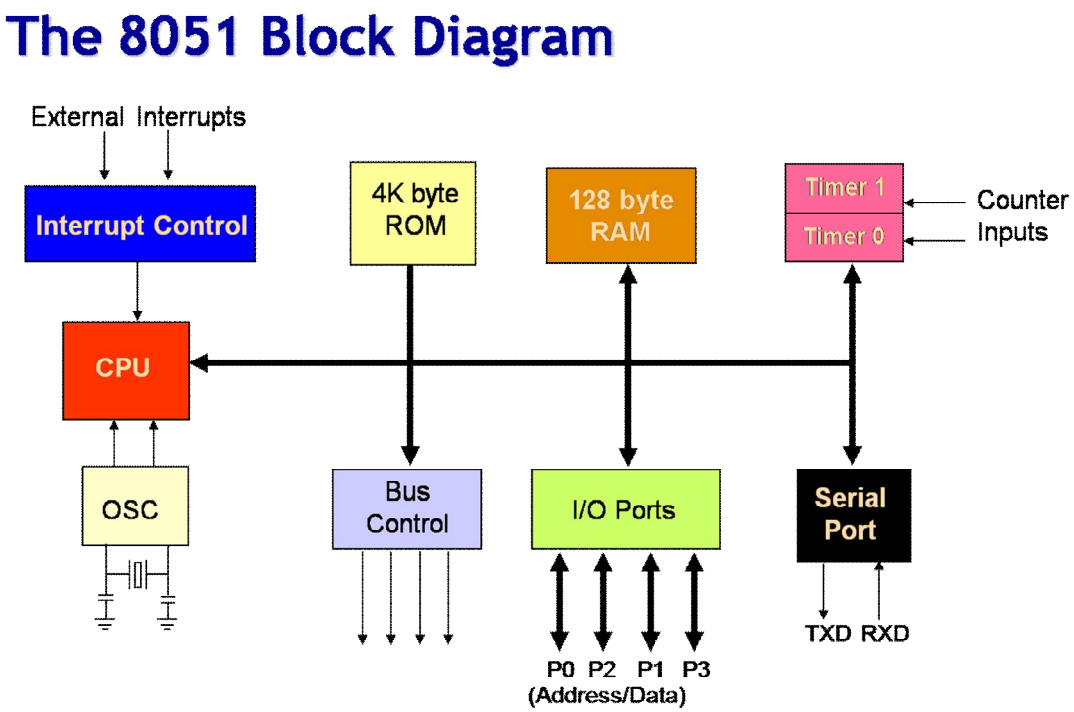
Bảng 1.2: Bảng so sánh đặc tính của 3 thành viên họ 8051

1.2. Kiến trúc phần cứng của họ 8051

1.2.1. Sơ đồ khối của 8051

Kiến trúc cơ bản bên trong 8051 bao gồm các khối chức năng sau

- CPU (Central Processing Unit): đơn vị điều khiển trung tâm
- Bộ nhớ chương trình ROM bao gồm 4 Kbyte
- Bộ nhớ dữ liệu RAM bao gồm 128 byte
- Bốn cổng xuất nhập
- Hai bộ định thời/bộ đếm 16 bit thực hiện chức năng định thời và đếm sự kiện
- Bộ giao diện nối tiếp (cổng nối tiếp)
- Khối điều khiển ngắt với hai nguồn ngắt ngoài
- Bộ chia tần số



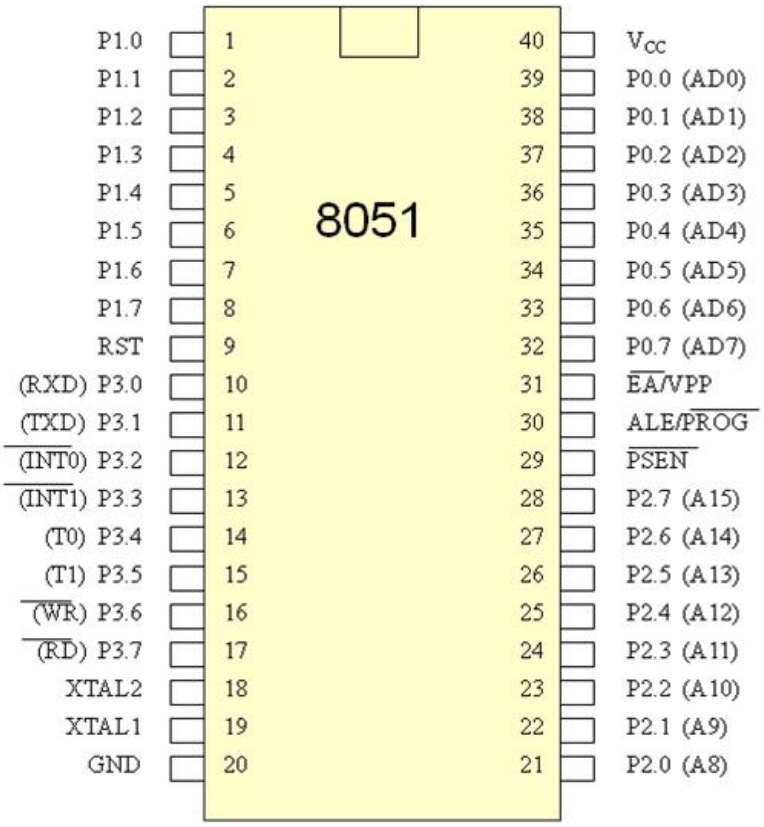
Hình 1.1: Sơ đồ khối của bộ vi điều khiển 8051

1.2.2. Sơ đồ chân và chức năng các chân của vi điều khiển 8051

Hình 1.2 cho ta sơ đồ chân của chip 8051, mô tả chức năng các chân như sau:

* ***Chân 1 đến 8***: được gọi là Cổng 1 (Port 1),

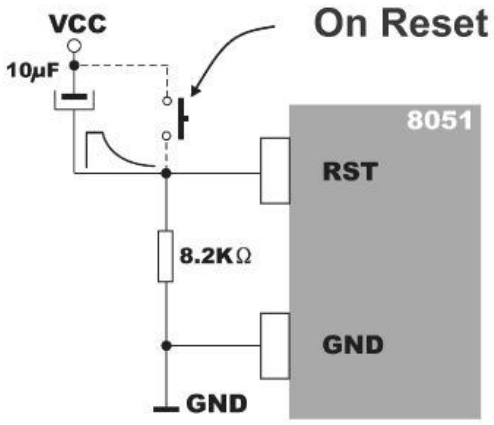
Tám chân này có duy nhất 1 chức năng là xuất và nhập. Cổng 1 có thể xuất và nhập theo bit hoặc byte. Ta đánh tên cho mỗi chân của Port 1 là P1.X (X = 0 đến 7)



Hình 1.2: Sơ đồ chân của 8051

* ***Chân 9***: là chân vào reset của 8051

Khi tín hiệu này được đưa lên mức cao trong ít nhất là **2 chu kỳ máy**, các thanh ghi trong bộ vi điều khiển được tải những giá trị thích hợp để khởi động hệ thống. Hay nói cách khác là vi điều khiển sẽ bị reset nếu chân này được kích hoạt mức cao.



Hình 1.3: Sơ đồ mạch reset ngoài của 8051

* ***Chân 10 đến 17***: được gọi là Cổng 3 (Port 3)

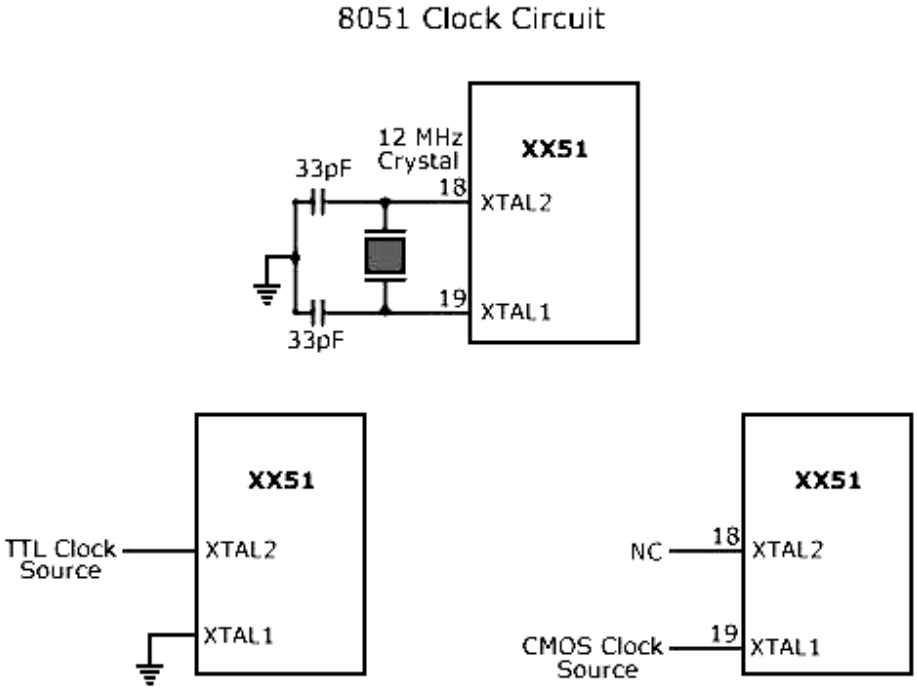
Tám chân này ngoài chức năng là xuất và nhập như các chân ở cổng 1 (chân 1 đến 8) thì mỗi chân này còn có chức năng riêng nữa, cụ thể như sau:

Bit	Tên	Chức năng
P3.0	RxD	Chân nhận dữ liệu cho cổng nối tiếp
P3.1	TxD	Chân truyền dữ liệu cho cổng nối tiếp
P3.2	INT0	Chân ngắt bên ngoài 0
P3.3	INT1	Chân ngắt bên ngoài 1
P3.4	T0	Ngõ vào của Timer/counter 0
P3.5	T1	Ngõ vào của Timer/counter 1
P3.6	WR	Xung ghi bộ nhớ dữ liệu ngoài
P3.7	RD	Xung đọc bộ nhớ dữ liệu ngoài

Bảng 1.3: Bảng mô tả chức năng riêng của cổng 3

* ***Chân 18 và 19*** (XTAL1 & XTAL2)

Hai chân này được sử dụng để nối với bộ dao động ngoài



Hình 1.4: Mạch dao động cấp cho 8051

Thông thường một bộ dao động thạch anh sẽ được nối tới các chân đầu vào XTAL1 (chân 19) và XTAL2 (chân 18) cùng với hai tụ gốm giá trị khoảng 30pF. Một phía của tụ điện được nối xuống đất như hình trên.

Các hệ thống xây dựng trên 8051 thường có tần số thạch anh từ 10 đến 40 MHz, thông thường ta dùng thạch anh 12 Mhz

* ***Chân 20***: được nối vào chân 0V của nguồn cấp

* ***Chân 21 đến chân 28***: được gọi là cổng 2 (Port 2)

Tám chân của cổng 2 có 2 công dụng, ngoài chức năng là cổng xuất và nhập như cổng 1 thì cổng 2 này còn là byte cao của bus địa chỉ khi sử dụng bộ nhớ ngoài.

* ***Chân 29*** (PSEN):

Chân PSEN là chân điều khiển đọc chương trình ở bộ nhớ ngoài, nó được nối với chân OE của ROM ngoài để cho phép đọc các byte mã lệnh trên ROM ngoài. PSEN ở mức thấp trong thời gian đọc mã lệnh.

Khi thực hiện chương trình trong ROM nội thì PSEN được duy trì ở mức cao

* ***Chân 30*** (ALE):

Chân ALE cho phép tách các đường dữ liệu và các đường địa chỉ tại Port 0 và Port 2.

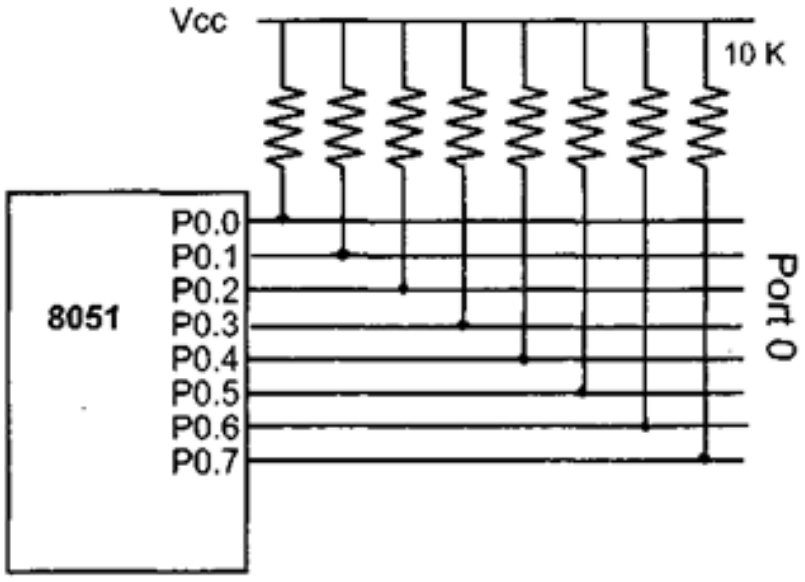
* ***Chân 31*** (EA):

Tín hiệu chân EA cho phép chọn bộ nhớ chương trình là bộ nhớ trong hay ngoài vi điều khiển. Nếu chân EA được nối ở mức cao (nối nguồn Vcc), thì vi điều khiển thi hành chương trình trong ROM nội. Nếu chân EA ở mức thấp (được nối GND) thì vi điều khiển thi hành chương trình từ bộ nhớ ngoài.

* ***Chân 32 đến 39***: được gọi là cổng 0 (Port 0)

Cổng 0 gồm 8 chân cũng có 2 công dụng, ngoài chức năng xuất nhập, cổng 0 còn là bus đa hợp dữ liệu và địa chỉ, chức năng này sẽ được sử dụng khi 8051 giao tiếp với các biết bị ngoài có kiến trúc Bus như các vi mạch nhớ...

Vì cổng P0 là một máng mở khác so với các cổng P1, P2 và P3 nên các chân ở cổng 0 phải được nối với điện trở kéo khi sử dụng các chân này như chân vào/ra. Điện trở này tùy thuộc vào đặc tính ngõ vào của thành phần ghép nối với chân của port 0. Thường ta dùng điện trở kéo khoảng 4K7 đến 10K



Hình 1.5: Nối điện trở kéo cho cổng 0 của 8051

* ***Chân 40***: chân nguồn của vi điều khiển, được nối vào chân Vcc của nguồn

1.3. Tổ chức bộ nhớ

Trên vi điều khiển 8051/8052 đều có cả bộ nhớ chương trình (ROM) và bộ nhớ dữ liệu (RAM). Tuy nhiên dung lượng của các bộ nhớ trên chip là hạn chế. Khi thiết kế các ứng dụng đòi hỏi bộ nhớ lớn người ta có thể dùng bộ nhớ ngoài.

1.3.1. Bộ nhớ chương trình

Bộ nhớ chương trình là bộ nhớ chỉ đọc, là nơi lưu trữ chương trình của vi điều khiển. Bộ nhớ chương trình của họ 8051 có thể thuộc một trong các loại sau ROM, EPROM, FLASH hoặc không có bộ nhớ chương trình trên chip. Với họ vi điều khiển 89xx, bộ nhớ chương trình được tích hợp sẵn trong chip có kích thước nhỏ nhất là 4kByte. Với các vi điều khiển không tích hợp sẵn bộ nhớ chương trình trên chip, buộc phải thiết kế bộ nhớ chương trình bên ngoài.

Địa chỉ đầu tiên của bộ nhớ chương trình là 0000H, chính là địa chỉ reset của vi điều khiển. Ngay khi bật nguồn hoặc reset vi điều khiển, thì CPU sẽ nhảy đến thực hiện lệnh ở địa chỉ 0000H này.

Khi sử dụng bộ nhớ trên chip thì chân EA phải được nối lên mức logic cao (+5V). Nếu bạn muốn mở rộng bộ nhớ chương trình thì chúng ta phải dùng bộ nhớ ngoài với dung lượng tối đa là 64Kbyte.

1.3.2. Bộ nhớ dữ liệu

Bộ nhớ dữ liệu tồn tại độc lập so với bộ nhớ chương trình. Họ vi điều khiển 8051 có bộ nhớ dữ liệu tích hợp trên chip nhỏ nhất là 128byte và có thể mở rộng với bộ nhớ dữ liệu ngoài lên tới 64kByte.

Bộ nhớ dữ liệu được phân chia như sau:

*** Các băng thanh ghi có địa chỉ từ 00H đến 1FH**

32 byte thấp của bộ nhớ nội được dùng cho các băng thanh ghi (dãy thanh ghi). Bộ lệnh 8051 hỗ trợ 8 thanh ghi R0 đến R7 và theo mặc định sau khi reset hệ thống, các thanh ghi này có các địa chỉ từ 00H đến 07H.

Do có 4 băng thanh ghi nên tại một thời điểm chỉ có duy nhất 1 băng thanh ghi được truy suất bởi các thanh ghi R0 - R7, để thay đổi các băng thanh ghi thì ta thay đổi các bit chọn băng trong thanh ghi trạng thái PSW.

*** Vùng RAM địa chỉ hóa từng bit có địa chỉ từ 20h đến 2Fh**

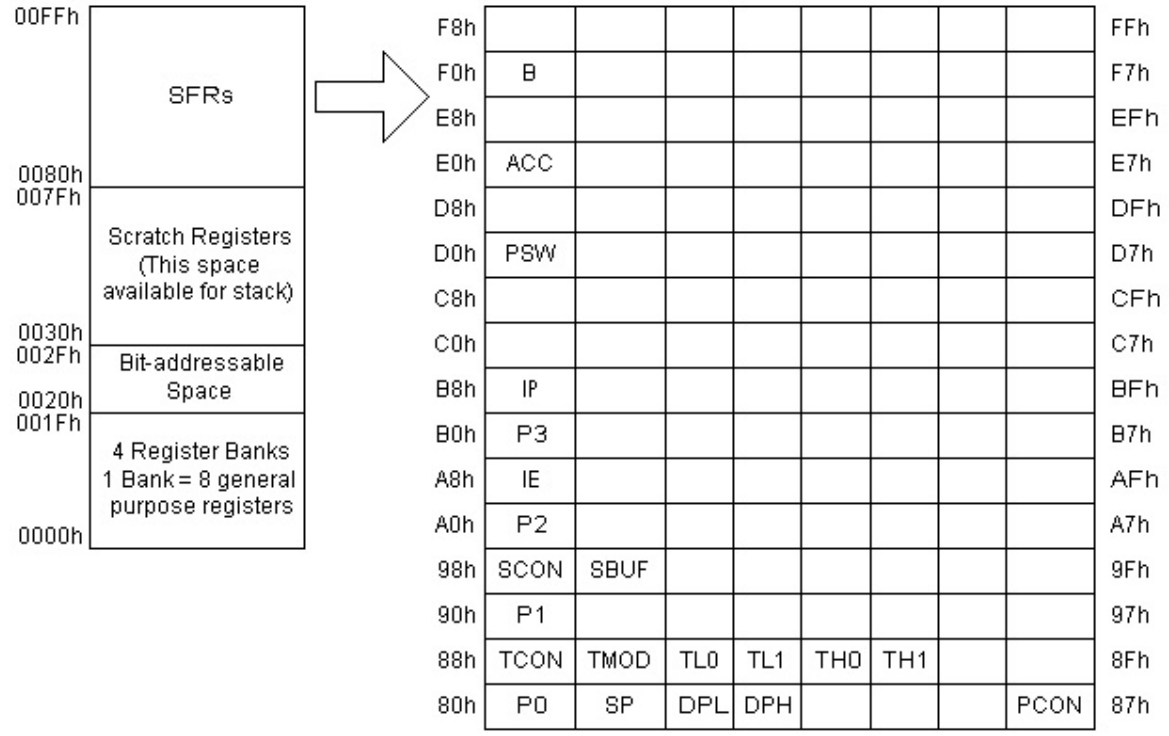
8051 chứa 210 vị trí bit được định địa chỉ trong đó 128 bit chứa trong các byte ở địa chỉ từ 20H đến 2FH (16 byte x 8 bit = 128 bit) và phần còn lại chứa trong các thanh ghi đặc biệt. Ngoài ra 8051 còn có các cổng xuất/nhập có thể định địa chỉ từng bit, điều này làm đơn giản việc giao tiếp bằng phần mềm với các thiết bị xuất/nhập đơn bit.

*** Vùng RAM đa dụng có địa chỉ từ 30h đến 7Fh**

Bất kỳ vị trí nhớ nào trong vùng RAM đa mục đích đều có thể được truy xuất tự do bằng cách sử dụng các kiểu định địa chỉ trực tiếp hoặc gián tiếp

*** Các thanh ghi chức năng đặc biệt có địa chỉ từ 80h đến FFh**

Cũng như các thanh ghi từ R0 đến R7, ta có 21 thanh ghi chức năng đặc biệt SFR chiếm phần trên của Ram nội từ địa chỉ 80H đến FFH. Cần lưu ý là không phải tất cả 128 địa chỉ từ 80H đến FFH đều được định nghĩa mà chỉ có 21 địa chỉ được định nghĩa.



Hình 1.6: Cấu trúc bộ nhớ dữ liệu của 8051

1.3.3. Các thanh ghi chức năng đặc biệt (SFR)

Thanh ghi của 8051 được dùng để lưu trữ tạm thời dữ liệu hoặc địa chỉ. Các thanh ghi này chủ yếu có kích thước 8 bit, 8 bit của các thanh ghi được sắp xếp như hình dưới trong đó bit D7 là bit có trọng số cao nhất, còn bit D0 là bit có trọng số thấp nhất.

D7	D6	D5	D4	D3	D2	D1	D0

Hình 1.7: Thanh ghi 8 bit

Phần dưới đây, chúng ta sẽ nghiên cứu khái quát về các thanh ghi cơ bản của 8051. Chi tiết hoạt động của mỗi thanh ghi, cờ... sẽ được tìm hiểu kỹ khi học lệnh lập trình.

a) Thanh ghi chính A

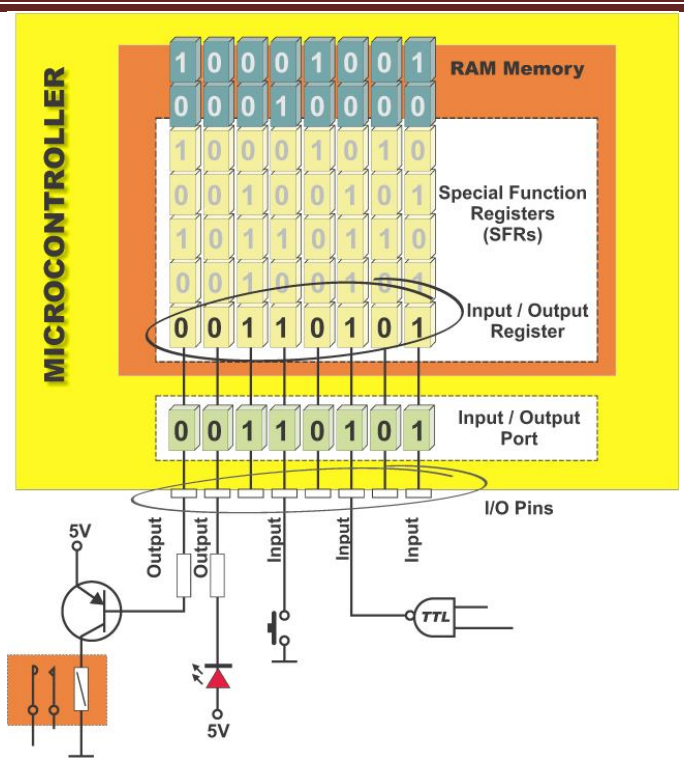
Là thanh ghi đặc biệt của 8051 dùng để thực hiện các phép toán của CPU, thường kí hiệu là A (Accumulator).

b) Thanh ghi phụ B

Là thanh ghi tính toán phụ của vi điều khiển 8051, ở địa chỉ F0H được dùng chung với thanh ghi chính A trong các phép toán nhân, chia. Thanh ghi B cũng được sử dụng như một thanh ghi trung gian

c) Thanh ghi cổng P0 - P3

Các port xuất/nhập của 8051 bao gồm Port 0 tại địa chỉ 80H, Port 1 tại địa chỉ 90H, Port 2 tại địa chỉ A0H và Port 3 tại địa chỉ B0H. Tất cả các port này đều có thể truy suất theo bit hoặc theo byte.



Hình 1.8: Mô tả chức năng xuất/nhập của thanh ghi cổng

d) Thanh ghi trạng thái chương trình PSW

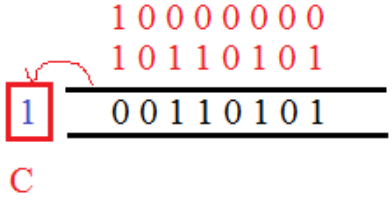
Thanh ghi trạng thái chương trình PSW (địa chỉ: D0H) là thanh ghi mô tả toàn bộ trạng thái chương trình đang hoạt động của hệ thống.

PSW.7	PSW.6	PSW.5	PSW.4	PSW.3	PSW.2	PSW.1	PSW.0
CY	AC	F0	RS1	RS0	OV	-	P

Bảng 1.4: Thanh ghi trạng thái chương trình PSW

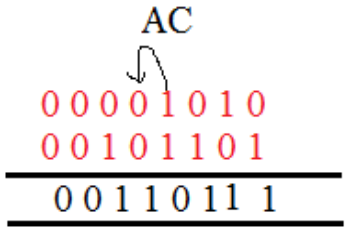
Giải thích chức năng từng bit của thanh trạng thái chương trình PSW

* **Cờ nhớ CY (Carry Flag) - địa chỉ D7h**: cờ nhớ có tác dụng kép (dùng trong các phép toán số học và logic). Thông thường nó được dùng cho các lệnh toán học: Cờ C được Set lên 1 nếu phép cộng có Bit nhớ từ Bit 7 (bị tràn) hoặc phép trừ có Bit mượn cho Bit 7. Ngược lại, cờ nhớ C = 0 nếu phép cộng không có nhớ và phép trừ không có mượn. Để dễ hình dung, ta xem ví dụ ở hình dưới đây



Hình 1.9: ví dụ về cờ nhớ CY

*** Cờ nhớ phụ AC (Auxiliary Carry Flag) - địa chỉ D6h**: khi cộng những giá trị BCD, cờ nhớ phụ được Set lên 1 nếu có Bit nhớ từ Bit 3 (bit thứ 4) sang Bit 4 hoặc kết quả trong 4 Bit thấp nằm trong khoảng 0AH->0FH.



Hình 1.10: ví dụ về cờ phụ AC

* **Cờ 0 (Flag 0) - địa chỉ D5h**: cờ 0 là một bit cờ đa dụng dùng cho các ứng dụng của người dùng

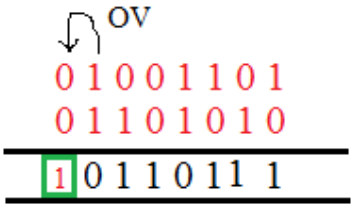
* **Bit chọn băng thanh ghi RS1 và RS0 - địa chỉ D4h & D3h**: hai bit này được dùng để chọn băng thanh ghi R0, R1, R2 hay R3. Chúng được xóa về 0 khi chip bị reset và được thiết lập mức 1 hay 0 bằng phần mềm.

Tùy theo RS1, RS0 có giá trị bằng 00,01,10 hay 11 sẽ chọn được băng thanh ghi tương ứng Băng 0, Băng 1, Băng 2, Băng 3

RS1	RS0	Register Bank
0	0	0
0	1	1
1	0	2
1	1	3

* **Cờ tràn OV (Overflow flag) - địa chỉ D2h:** cờ tràn được thiết lập (OV = 1) sau một hoạt động cộng hoặc trừ nếu có sự tràn toán học.

- Khi các số có dấu được cộng hoặc trừ với nhau, ta có thể dùng cờ này để kiểm tra xem kết quả có nằm trong giới hạn xác định không (-127, +128).
- Khi cộng hoặc trừ các số không dấu thì cờ này được bỏ qua.



Hình 1.11: ví dụ về cờ tràn OV

*** Bit PWS.1 - địa chỉ D1h**: bit được dùng cho người dùng định nghĩa

*** Cờ chẵn lẻ P - địa chỉ D0h**: phản ánh số bit 1 trong thanh ghi A là chẵn hay lẻ. Nếu thanh ghi A chứa một số chẵn các bít 1 thì P = 0 còn chứa một số lẻ bit 1 thì P = 1.

e) Con trỏ ngăn xếp SP (Stack Point)

Ngăn xếp chính là vùng bộ nhớ RAM được CPU sử dụng để lưu thông tin tạm thời. Thông tin này có thể là dữ liệu hoặc địa chỉ. CPU cần các thanh ghi này vì số các thanh ghi bị hạn chế.

Như vậy, để có thể truy cập vào vùng nhớ ngăn xếp thì cần phải có thanh ghi trong CPU trỏ đến. Thanh ghi SP này sẽ được dùng để trỏ đến ngăn xếp, nên được gọi là thanh ghi con trỏ ngăn xếp.Thanh ghi này có độ rộng là 8 bit, tức là chỉ có thể trỏ được các địa chỉ từ 00h đến FFh.

f) Con trỏ dữ liệu DPTR (Data pointer)

Con trỏ dữ liệu được dùng để truy xuất bộ nhớ chương trình ngoài hoặc bộ nhớ dữ liệu ngoài. Con trỏ dữ liệu là một thanh ghi 16 bit ở địa chỉ 82H (DPL - byte thấp) và 83H (DPH - byte cao).

g) Thanh ghi bộ đệm truyền thông nối tiếp SBUF (Serial Data Buffer)

Bộ đệm truyền thông được chia thành hai bộ đệm, bộ đệm truyền dữ liệu và bộ đệm nhận dữ liệu. Khi dữ liệu được chuyển vào thanh ghi SBUF, dữ liệu sẽ được chuyển vào bộ đệm truyền dữ liệu và sẽ được lưu giữ ở đó cho đến khi quá trình truyền dữ liệu qua truyền thông nối tiếp kết thúc. Khi thực hiện việc chuyển dữ liệu từ SBUF ra ngoài, dữ liệu sẽ được lấy từ bộ đệm nhận dữ liệu của truyền thông nối tiếp.

h) Thanh ghi của bộ định thời/bộ đếm

8051 có 2 bộ đếm/định thời (counter/timer) 16 bit để định các khoảng thời gian hoặc để đếm các sự kiện. Các cặp thanh ghi (TH0, TL0) và (TH1, TL1) là các thanh ghi của bộ đếm thời gian. Bộ định thời 0 có địa chỉ 8AH (TL0, byte thấp) và 8CH (TH0, byte cao). Bộ định thời 1 có địa chỉ 8BH (TL1, byte thấp) và 8DH (TH1, byte cao).

Hoạt động của bộ định thời được thiết lập bởi thanh ghi chế độ định thời TMOD (Timer Mode Register) ở địa chỉ 88H. Chỉ có TCON được định địa chỉ từng bit.

i) Thanh ghi ngắt (Interrupt register)

Ngắt (Interrupt) - như tên của nó, là một số sự kiện khẩn cấp bên trong hoặc bên ngoài bộ vi điều khiển xảy ra, buộc vi điều khiển tạm dừng thực hiện chương trình hiện tại, phục vụ ngay lập tức nhiệm vụ mà ngắt yêu cầu – nhiệm vụ này gọi là trình phục vụ ngắt (ISR: Interrupt Service Routine).

Để thiết lập ngắt, ta sẽ thiết lập các giá trị trong thanh ghi cho phép ngắt IE ở địa chỉ A8H, thứ tự ưu tiên ngắt được đặt bằng cách set các bit ở thanh ghi ưu tiên ngắt IP ở địa chỉ B8h. Cả hai thanh ghi này được định địa chỉ theo bit.

1.4. Giới thiệu về lập trình hợp ngữ 8051 và phần mềm Kiel C

Trong phần này chúng ta bàn về dạng thức của hợp ngữ và định nghĩa một số thuật ngữ sử dụng rộng rãi gắn liền với lập trình hợp ngữ.

CPU chỉ có thể làm việc với các số nhị phân và có thể chạy với tốc độ rất cao. Tuy nhiên, thật là ngán ngậm và chậm chạp đối với con người phải làm việc với các số 0 và 1 để lập trình cho máy tính. Một chương trình chứa các số 0 và 1 được gọi là ngôn ngữ máy.

Trong những ngày đầu của máy tính, các lập trình viên phải viết mã chương trình dưới dạng ngôn ngữ máy. Mặc dù hệ thống thập lục phân (số Hex) đã được sử dụng như một

cách hiệu quả hơn để biểu diễn các số nhị phân thì quá trình làm việc với mã máy vẫn còn là công việc cồng kềnh đối với con người. Cuối cùng, hợp ngữ cũng được xây dựng, trong đó có sử dụng các từ gợi nhớ cho các lệnh mã máy cộng với những đặc tính khác giúp cho việc lập trình nhanh hơn và ít mắc lỗi hơn. Thuật ngữ *từ gợi nhớ* (mnemonic) thường được sử dụng trong tài liệu khoa học và kỹ thuật máy tính để tham chiếu cho các mã lệnh và từ viết tắt tương đối dễ nhớ. Các chương trình hợp ngữ phải được dịch ra thanh mã máy bằng một chương trình được gọi là trình hợp ngữ (hợp dịch). Hợp ngữ được coi như là một ngôn ngữ bậc thấp vì nó giao tiếp trực tiếp với cấu trúc bên trong của CPU. Để lập trình trong hợp ngữ, lập trình viên phải biết tất cả các thanh ghi của CPU và kích thước của chúng cũng như các chi tiết khác.

Ngày nay, ta có thể sử dụng nhiều ngôn ngữ lập trình khác nhau, chẳng hạn như Basic, Pascal, C, C ++, Java và vô số ngôn ngữ khác. Các ngôn ngữ này được coi là những ngôn ngữ bậc cao vì lập trình viên không cần phải tương tác với các chi tiết bên trong của CPU. Một trình hợp dịch được dùng để dịch chương trình hợp ngữ ra mã máy còn (còn đôi khi cũng còn được gọi mà đối tượng (Object Code) hay mã lệnh (Opcode), còn các ngôn ngữ bậc cao được dịch thành các ngôn ngữ mã máy bằng một chương trình gọi là trình biên dịch.

1.4.1 Giới thiệu về hợp ngữ

Một chương trình hợp ngữ bao gồm một chuỗi các dòng lệnh hợp ngữ. Một lệnh hợp ngữ có chứa một từ gợi nhớ và tùy theo từng lệnh và sau nó có một hoặc hai toán hạng. Các toán hạng là các dữ liệu cần được thao tác và các từ gợi nhớ là các lệnh đối với CPU nói nó làm gì với các dữ liệu. Dưới đây là ví dụ về một chương trình hợp ngữ.

```
        ORG 0H ;          Bắt đầu (origin) tại ngăn nhớ 0
        MOV R5, #25H ;    Nạp 25H vào R5
        MOV R7, #34H ;    Nạp 34H vào R7
        MOV A, #0 ;       Nạp 0 vào thanh ghi A
        ADD A, R5 ;       Cộng nôi dụng R5 vào A (A = A + R5)
        ADD A, R7 ;       Cộng nội dung R7 vào A (A = A + R7)
        ADD A, #121H ;    Cộng giá trị 12H vào A (A = A + 12H)
HERE:   SJMP HERE ;       ở lại trong vòng lặp này
        END ;             Kết thúc tệp nguồn hợp ngữ
```

Như vậy cấu trúc của một chương trình hợp ngữ ta đã được biết, câu hỏi đặt ra là chương trình sẽ được tạo ra và hợp dịch như thế nào và làm thế nào để có thể chạy được?

Các bước để tạo ra một chương trình hợp ngữ có thể chạy được là:

1. Trước hết ta sử dụng một **trình soạn thảo** để viết được chương trình giống như chương trình ở ví dụ trên. Có nhiều trình soạn thảo tuyệt vời để soạn thảo chương trình như trình soạn thảo EDIT của MS-DOS hoặc Notepad của. Lưu ý rằng, trình soạn thảo phải có khả năng tạo ra tệp mã ASCII. Đối với nhiều trình hợp ngữ thì các tên tệp tuân theo các quy ước thường lệ của DOS, nhưng phần mở rộng của các tệp nguồn phải là **.asm** hay **.src** (source - tệp nguồn) tuỳ theo trình hợp ngữ mà ta sử dụng.

2. Tệp nguồn có phần mở rộng **.asm** được tạo ra ở bước 1 sẽ được nạp vào **trình hợp dịch** của 8051. Trình hợp dịch chuyển các lệnh ra mã máy. Trình hợp dịch sẽ tạo ra một tệp đối tượng và một tệp liệt kê với các thành phần mở rộng **.obj** và **.lst** tương ứng.

3. Các trình hợp dịch yêu cầu một bước thứ ba gọi là liên kết. Chương trình liên kết lấy một hoặc nhiều tệp đối tượng và tạo ra một tệp đối tượng tuyệt đối với thành phần mở rộng **.abs**.

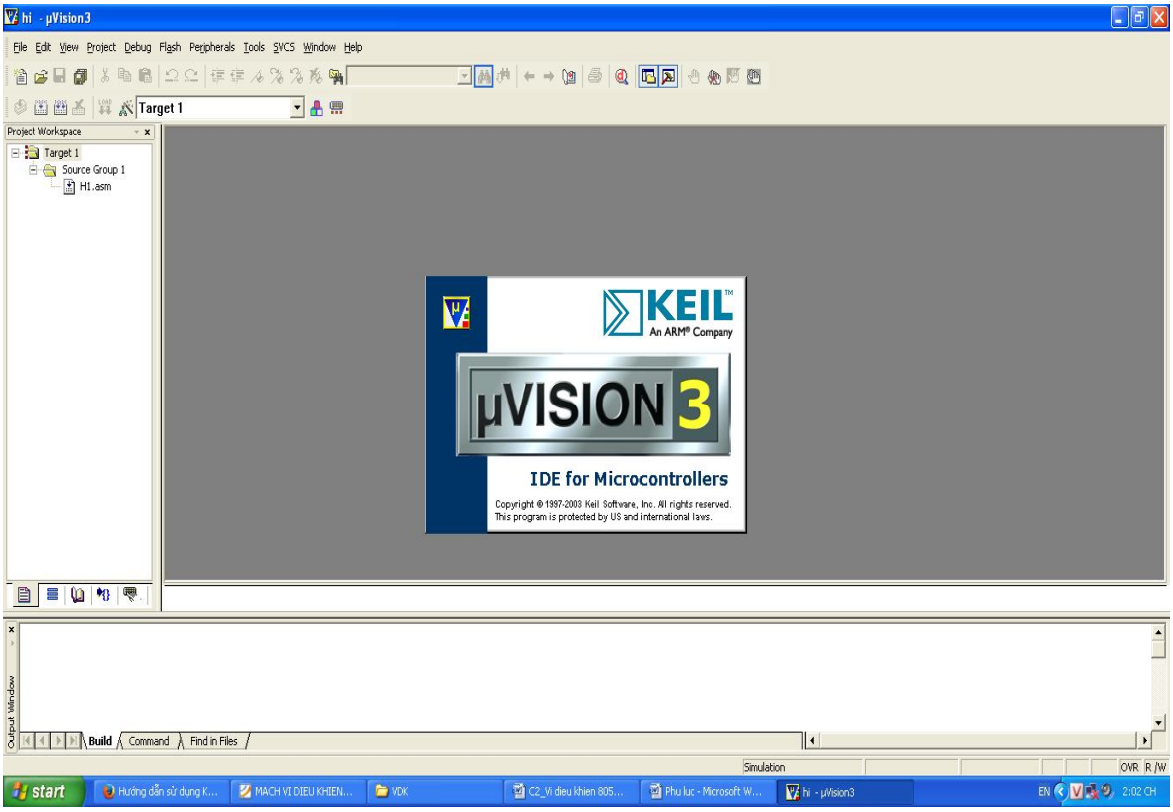
4. Kế sau đó tệp **.abs** được nạp vào một chương trình chuyển đổi tệp đối tượng về dạng số HEX được gọi là "OH". Tệp này đuôi mở rộng **Hex** và có thể nạp tốt vào trong ROM.

Hiện nay các phần mềm thường kết hợp các bước 2 đến 4 vào thành một bước để tạo được tệp .Hex. Điển hình như phần mềm **KEIL C**, đây là chương trình hỗ trợ khá đầy đủ trong việc lập trình cho vi điều khiển họ 8051 ngoài việc biên dịch bằng ngôn ngữ C bạn cũng có thể biên dịch dưới dạng ASM.

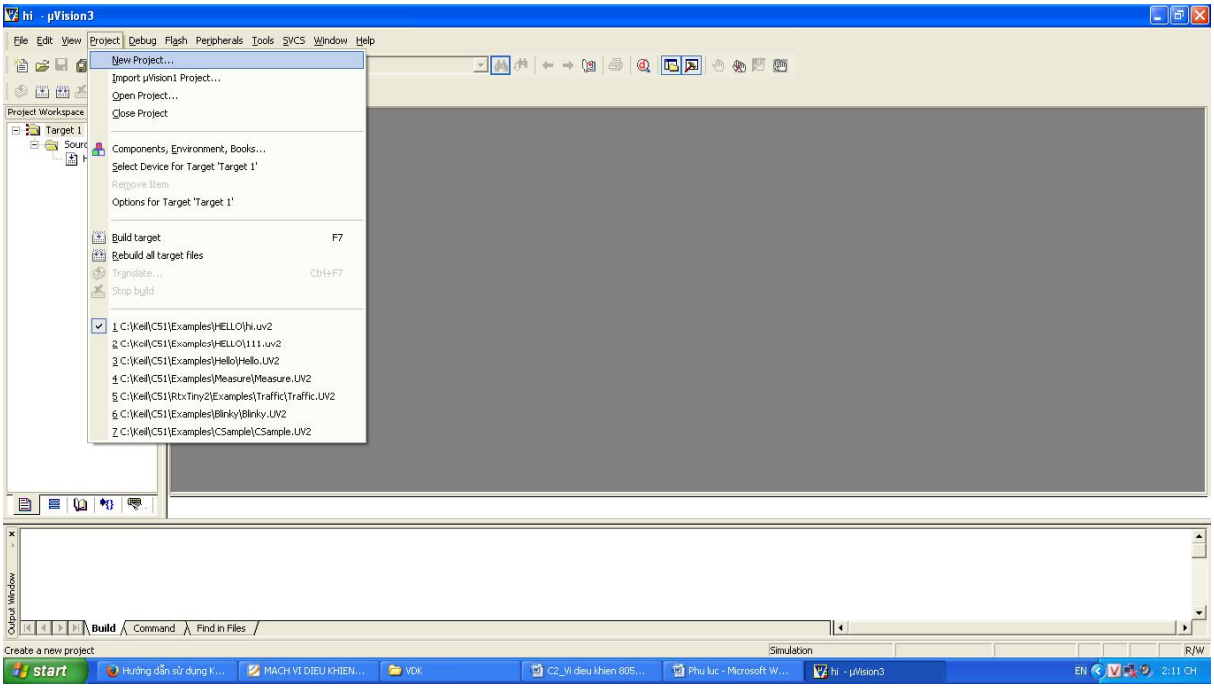
Trong phần mềm này, ta có thể soạn thảo chương trình bằng ngôn ngữ C hoặc hợp ngữ Assembly, sau đó sẽ dùng phần mềm này để biên dịch chương trình thành tệp Hex để nạp vào Chip.

1.4.2. Hướng dẫn sử dụng phần mềm KEIL C để soạn thảo và biên dịch chương trình

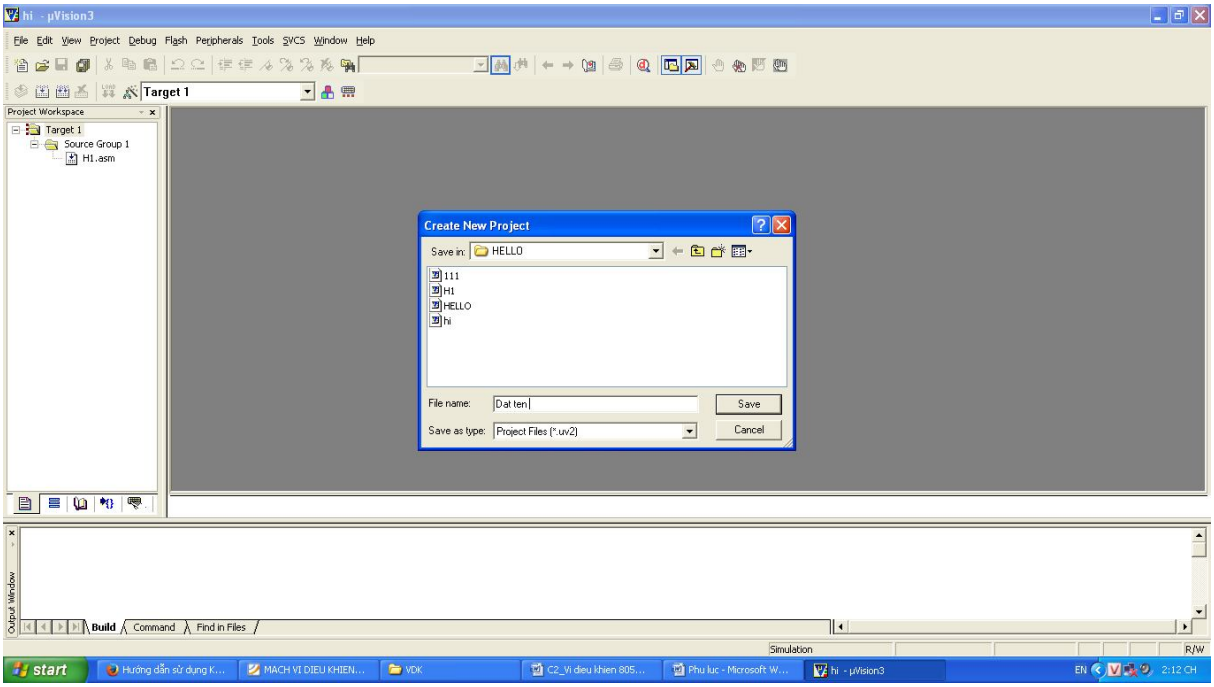
Keil C là công cụ phần mềm hỗ trợ khá đầy đủ để người dùng soạn thảo và biên dịch chương trình dành cho các vi điều khiển thuộc họ 8051 bằng ngôn ngữ C và Assembly.
Khi ta khởi động chương trình thì sẽ có giao diện như sau



Bước 1: Tạo một dự án - Vào Project -> Chọn new project

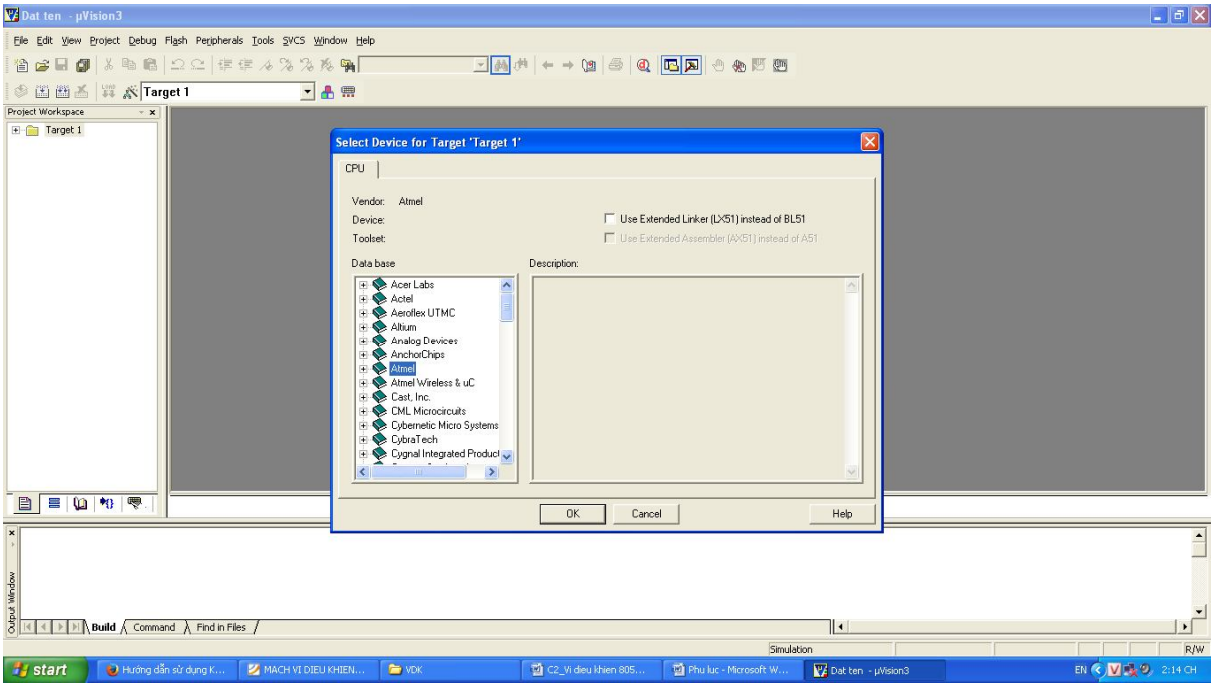


Bước 2: Chọn đường dẫn lưu Project, gõ tên Project vào khung File name, chọn Save

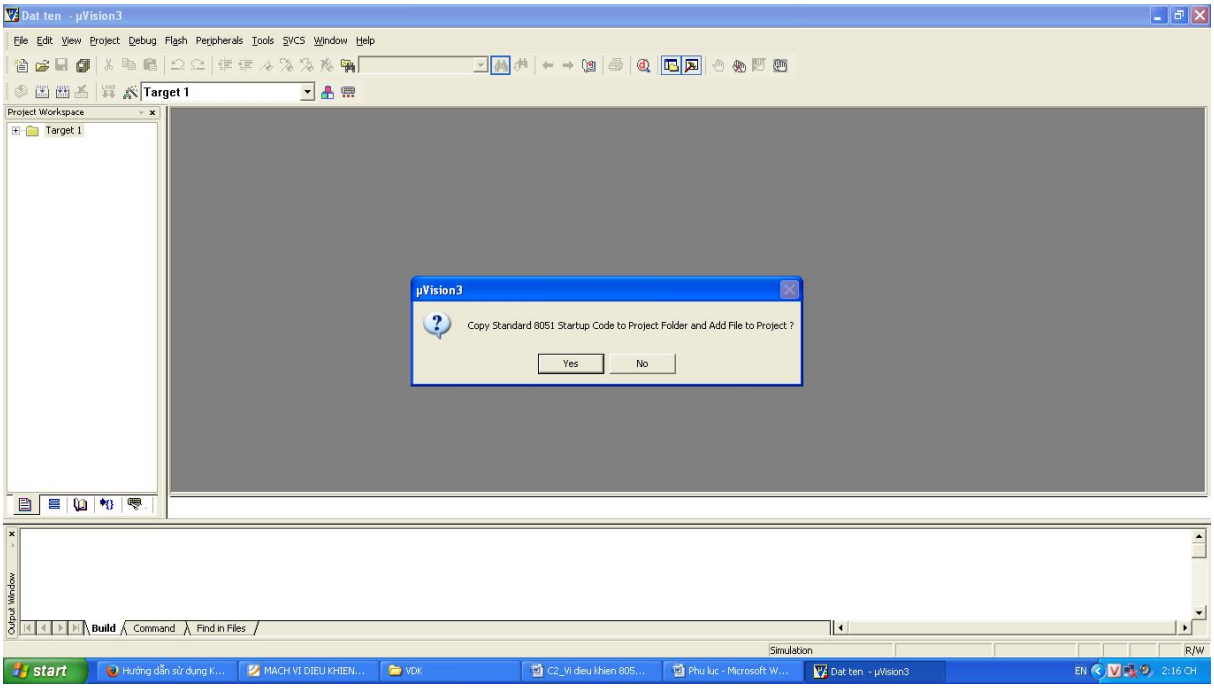


Bước 3: Chọn đến hãng sản xuất và tên chip cần lập trình

Ví dụ: chip AT89C51 của hãng Atmel, chọn OK

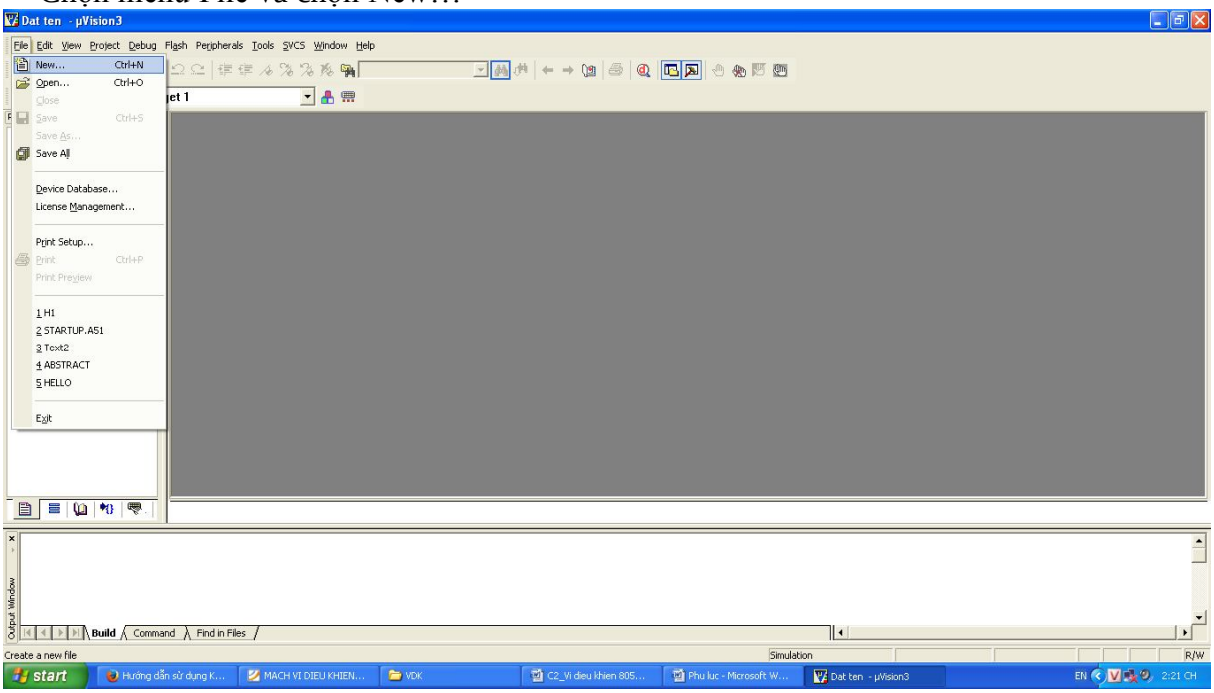


Bước 4: ta chọn Yes để hoàn thành việc tạo project



Bước 5: tạo file để viết chương trình.

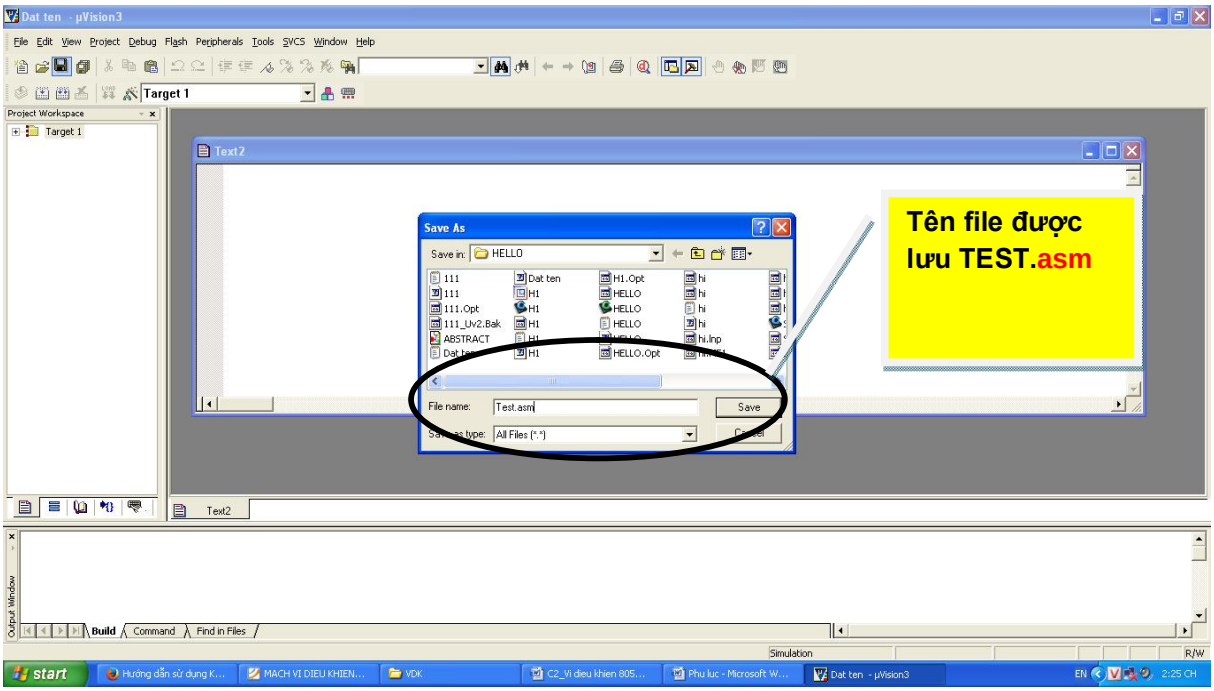
Chọn menu File và chọn New…



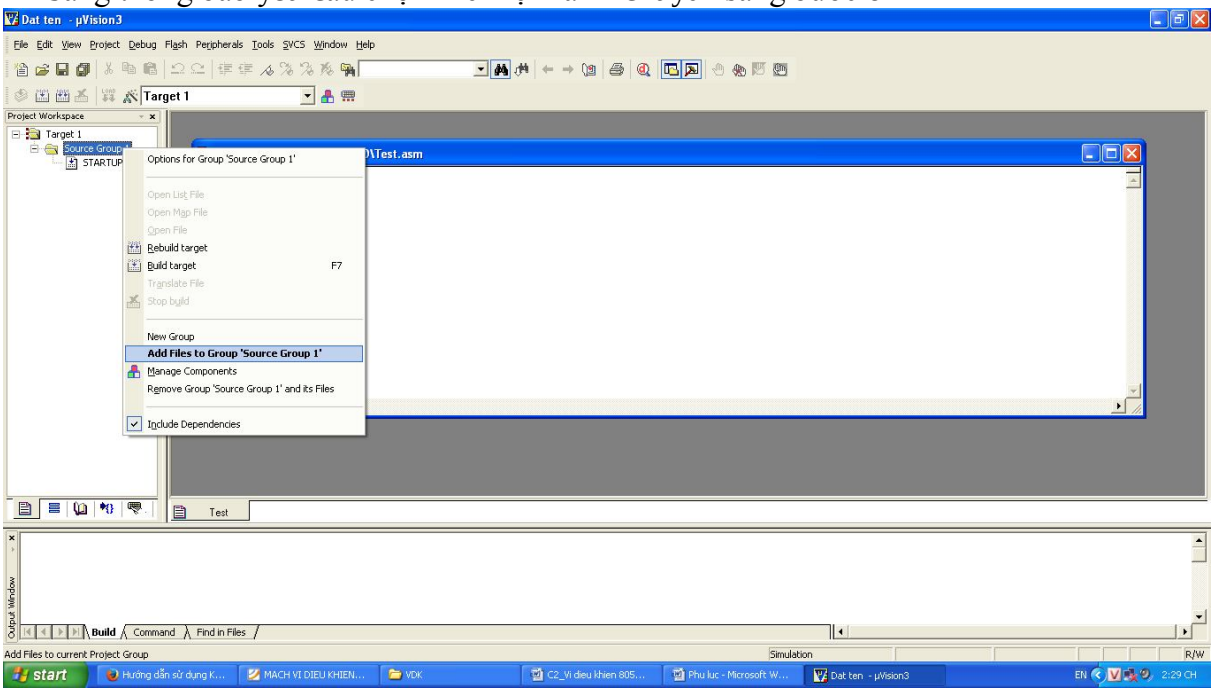
Bước 6: Chọn File -> Save -> chọn thư mục lưu Project ở trên và đặt tên cho file.

Lưu ý: Khi lập trình trên Keil C bằng ngôn ngữ **Assembly** phải thực hiện lưu file ở dạng có phần mở ".asm", ví dụ "Test.asm".

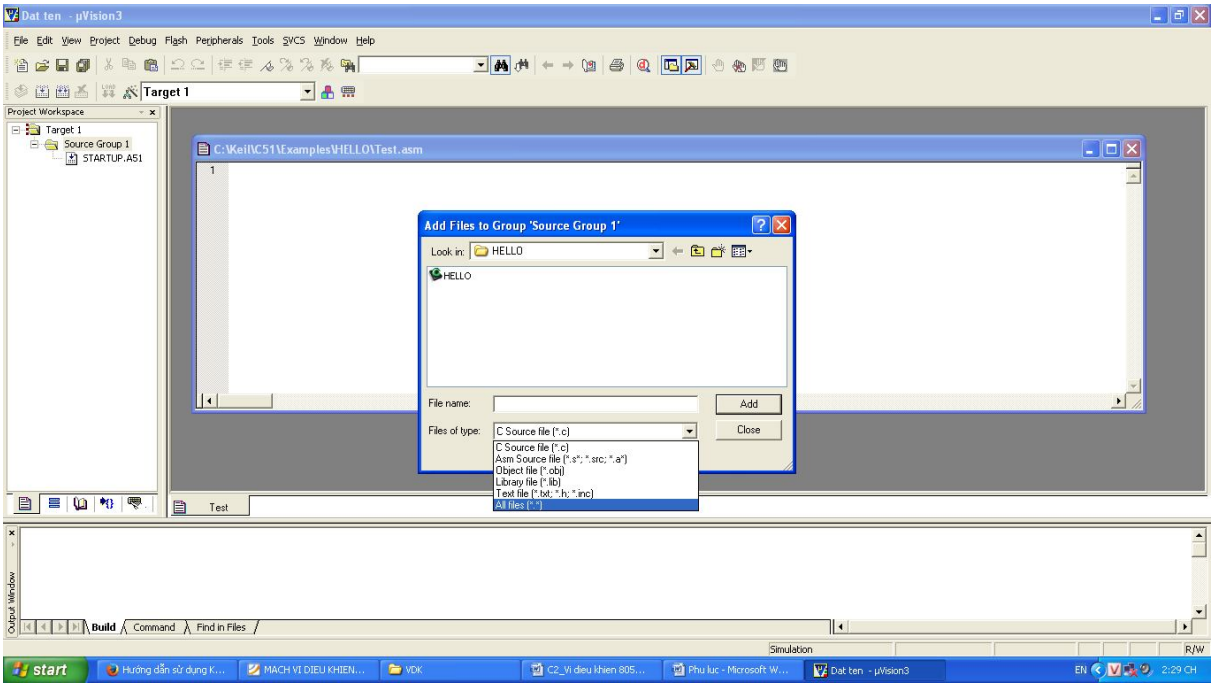
Nếu ta lập trình bằng ngôn ngữ C, thực hiện lưu file ở dạng file phần mở rộng ".c", ví dụ "Test.c".



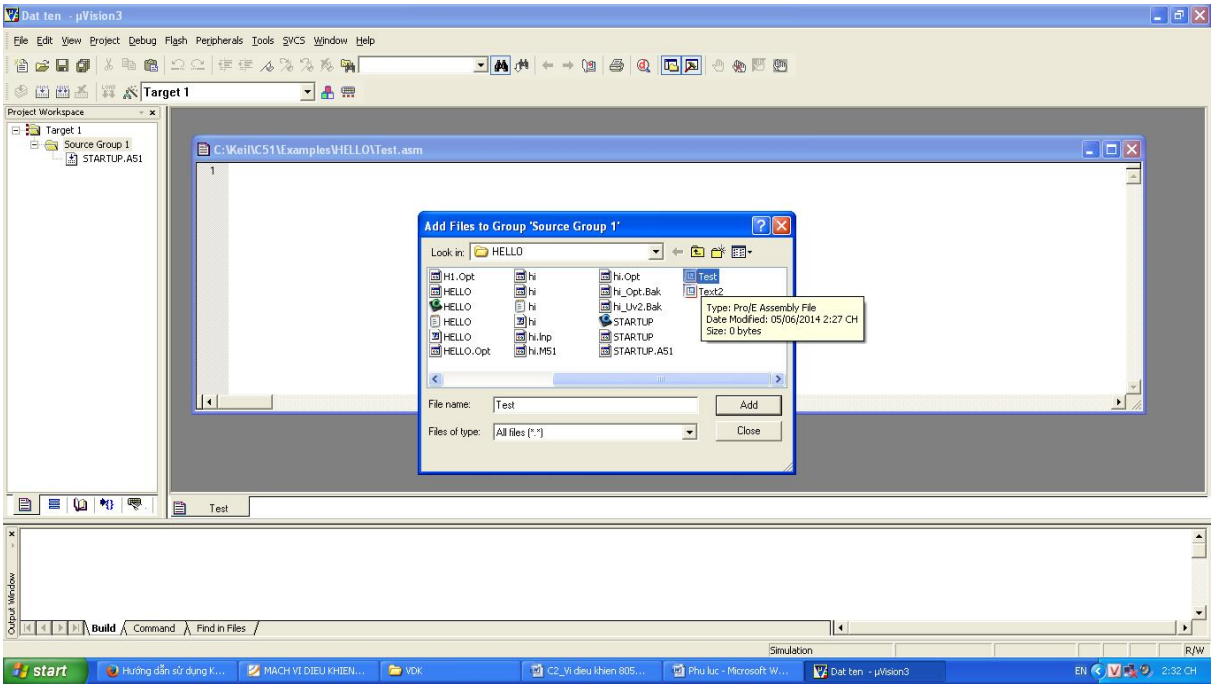
Bước 7: ta phải add file vừa tạo vào project bằng cách:
- Kích phải chuột vào SOURCE GROUP 1
- Chọn Add files to Group "Source Group 1"
- 1 bảng thông báo yêu cầu chọn file hiện ra -> chuyển sang bước 8



Bước 8: Chọn ALL FILE (*.*) ở phần File of type

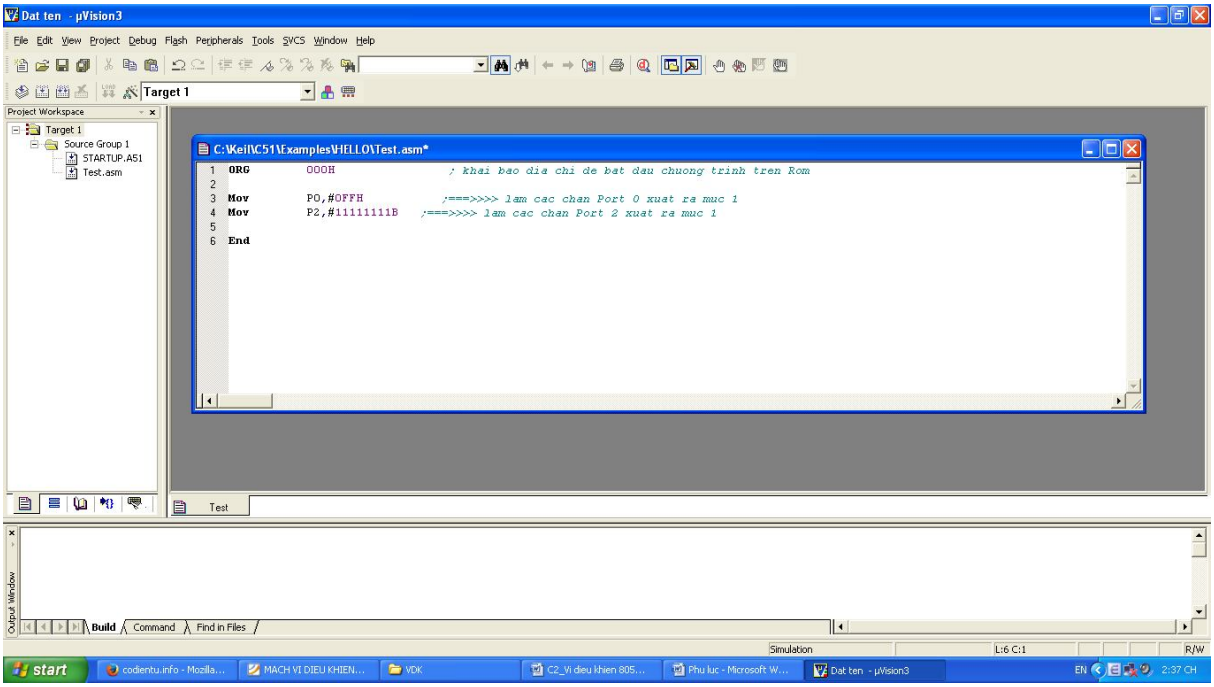


Bước 9: chọn file TEST.asm mà ta tạo ở bước trên và kích add -> chọn Close



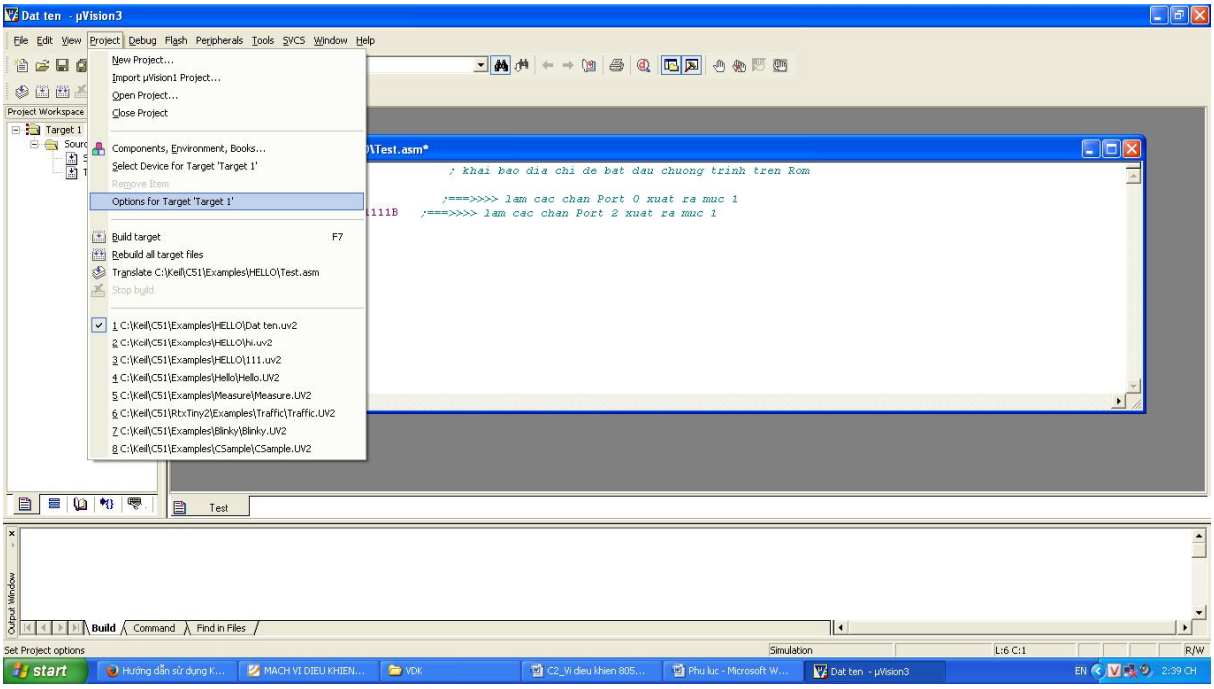
Như vậy ta đã hoàn thành việc tạo dự án - bây giờ ta bắt đầu soạn thảo chương trình.

Bước 10: Soạn thảo chương trình như hình dưới

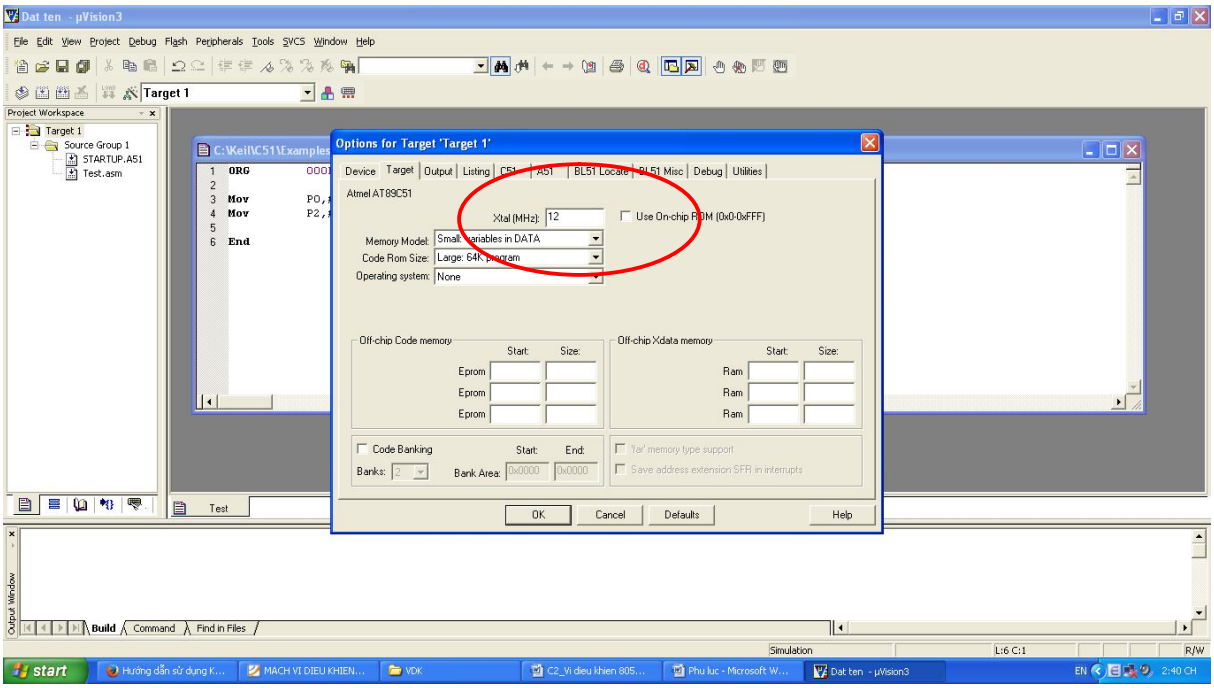


Bước 11: Sau khi hoàn thành việc soạn thảo chương trình, ta tiến hành dịch file asm này sang file hex để nạp vào chip

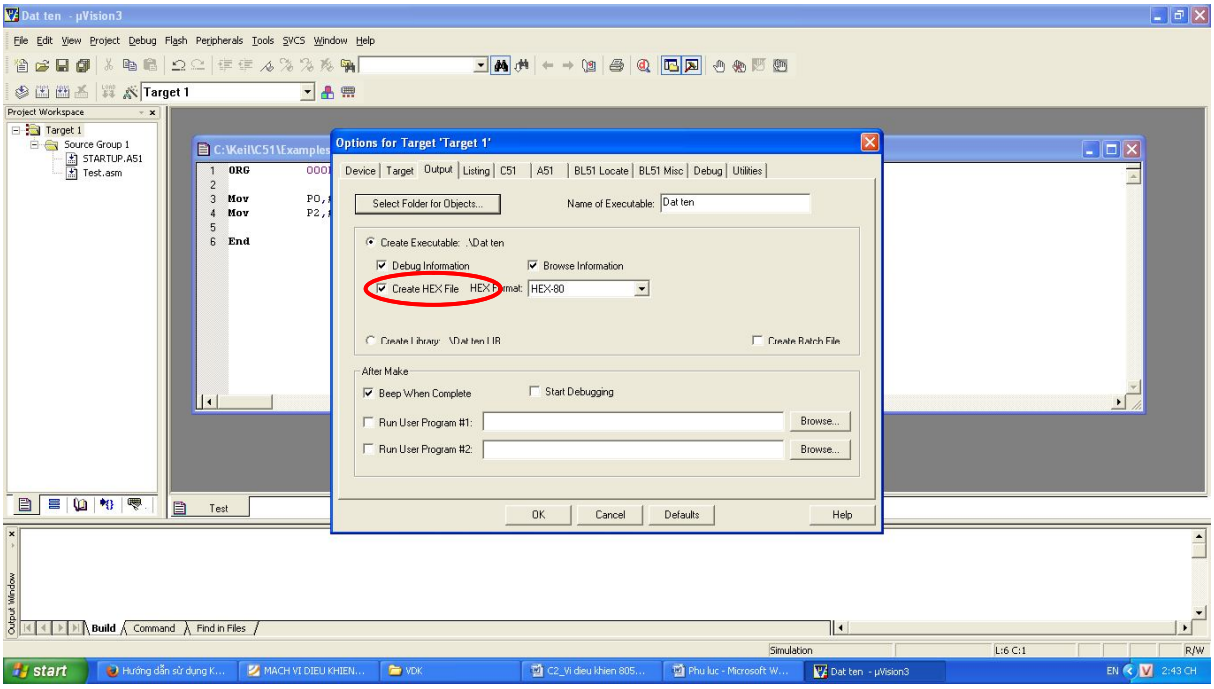
Ta chọn Project -> Options for Target "Target 1"



Bước 12: ta nhập tần số thạch anh mà lắp đặt cho vi điều khiển 8051. Ví dụ tôi nhập 12

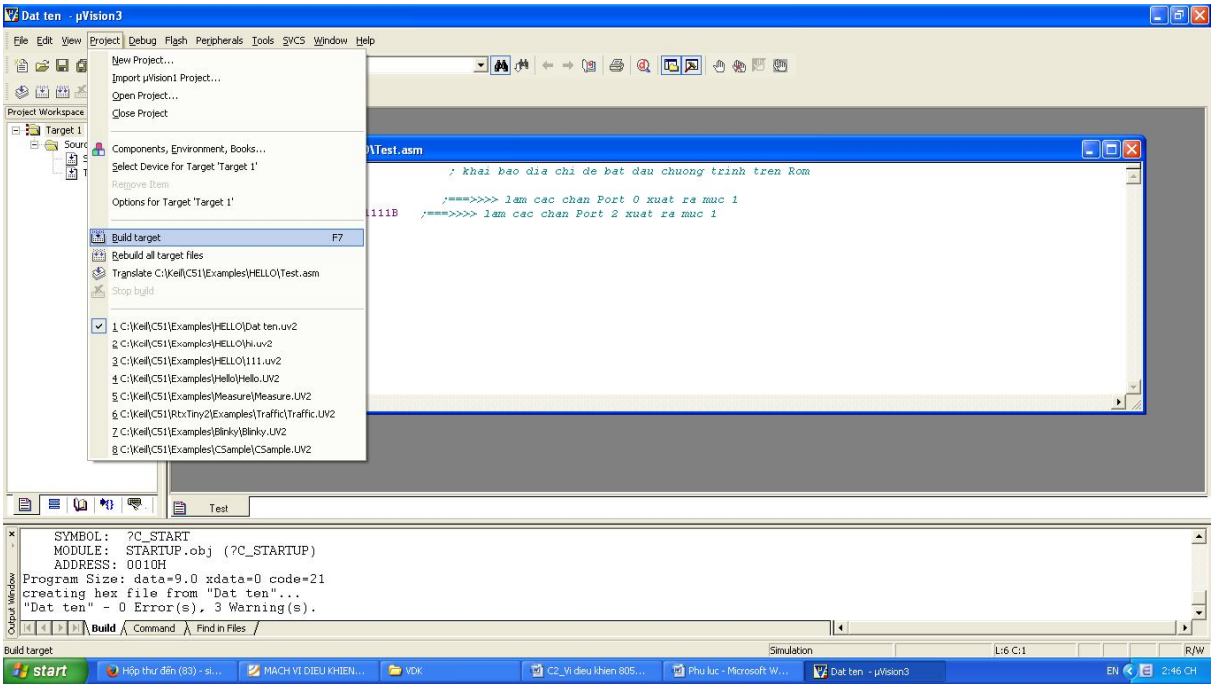


Bước 13: Chọn tab Output và click chọn Create HEX File, OK

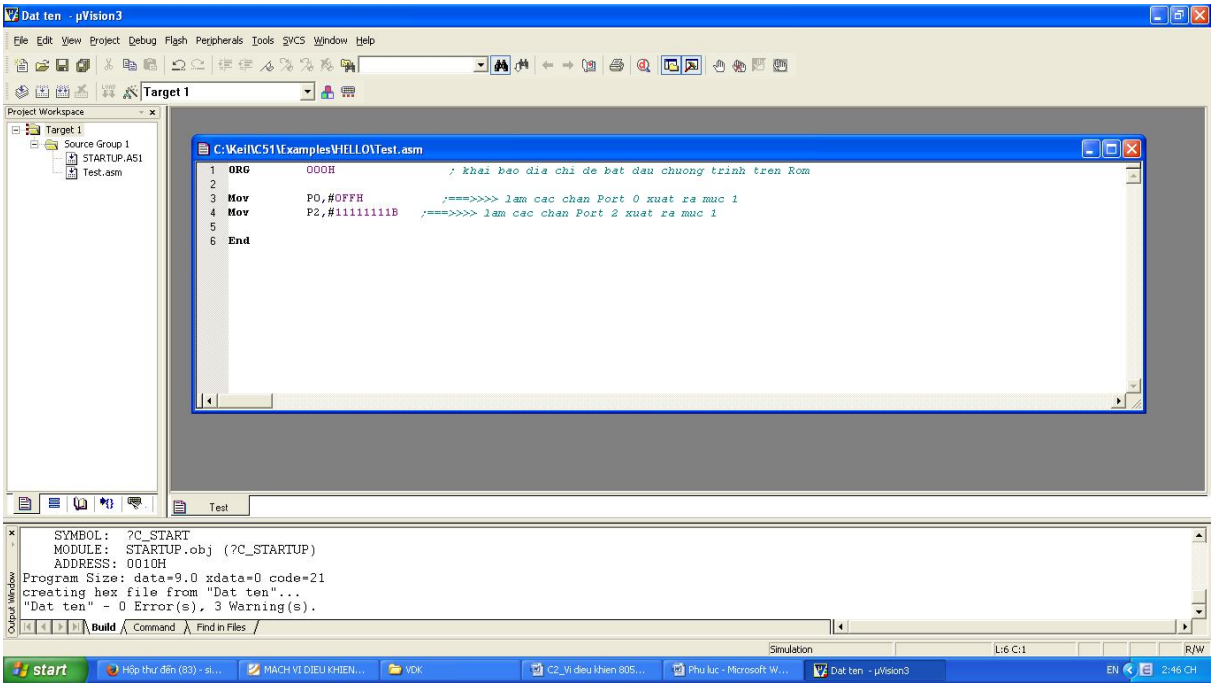


Bước 14: tạo file hex

Ta chọn Project -> Build Target F7



Bước 15: thông báo đã tạo file hex hay chưa, thông báo lỗi và cảnh báo



Như vậy ta đã hoàn thành việc chuẩn bị, soạn thảo và biên dịch chương trình. Ta sẽ sử dụng file .hex này để nạp vào chip 8051.