Using Internal EEPROM of PIC Microcontroller

18
Jun

Using Internal EEPROM of PIC Microcontroller

By Ligo George (https://electrosome.com/author/lijoppans/)

MikroC (https://electrosome.com/category/tutorials/pic-microcontroller/mikroc/), PIC Microcontroller (https://electrosome.com/category/tutorials/pic-microcontroller/), Tutorials (https://electrosome.com/category/tutorials/)

EEPROM (https://electrosome.com/tag/eeprom/), Microcontroller (https://electrosome.com/tag/microcontroller/), PIC (https://electrosome.com/tag/pic/), Tutorials (https://electrosome.com/tag/tutorials/)

32 Comments (https://electrosome.com/internal-eeprom-pic-microcontroller/#disqus_thread)

Contents

(https://electrosome.com/wp-content/uploads/2012/06/PIC16F877A.jpg)

There are commonly three types of memories in a PIC Microcontroller, Flash Program Memory, Data Memory (RAM) and EEPROM Data Memory. We write Programs in the Flash Program Memory of a microcontroller. Flash memory makes it possible to program a microcontroller many times before installing to device and even after the installation we can change the program. RAM Data Memory is used for storing data temporarily during program execution and it is volatile. That is, this memory is cleared when the power is gone or after CPU reset. RAM Data Memory locations are also called General Purpose Registers (GPR). These two memories have faster response time. The third memory is EEPROM memory which is an abbreviation for Electrically Erasable Programmable Read Only Memory. EEPROM memory can be read and write electrically, can be accessed through program. It is a non volatile memory but has slower response time. EEPROM memory can be used to store data such as sensor logs, device parameters which should not be loss during power loss or CPU reset. Here I using PIC 16F877A for explanation.

EEPROM Special Function Registers

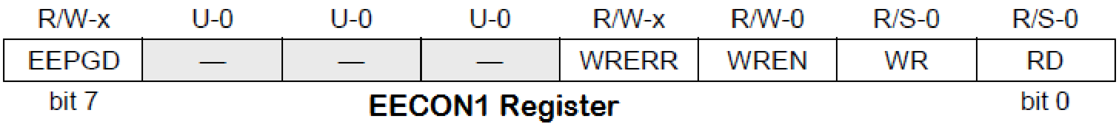
The data in the EEPROM and Flash Program Memory can be read/write during normal operations (over full VDD range). These memories are not mapped in the register file space, instead of it can be accessed through the following six Special Function Registers (SFR) for read and write operations.

- EECON1
- EECON2
- EEDATA
- EEDATH
- EEADR
- EEADRH

EEDATA register hold 8-bit data for read/write and EEADR holds the address of EEPROM memory location to be accessed. PIC Microcontrollers usually have 128/256 bytes of data EEPROM memory with address ranging from 00h to FFh. On devices having 128 bytes, memory locations from 80h to FFh are unimplemented and will be wraparound to beginning of the data EEPROM memory. On-Chip charge pump (used while writing data to EEPROM) is turned off when writing these unimplemented locations. EEDATH and EEADRH registers are used when interfacing program memory block with Program Flash Memory. Here we deals only with EEPROM data memory.

EEPROM data memory allows only single byte read and write. When a data byte is written in to EEPROM, automatically erases the particular location first and then writes the new data (erase before write). The write time is controlled by and on-chip timer and write/erase voltages are generated automatically by an on-chip charge pump. When the device is code protected, program or data memory will not be accessible to programmer but CPU may read or write data EEPROM memory.

EECON1 is the control register used for memory access.



R/W-x	U-0	U-0	U-0	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	—	—	—	WRERR	WREN	WR	RD
bit 7							bit 0

EECON1 Register

(https://electrosome.com/wp-content/uploads/2012/06/EECON1-register-PIC-Microcontroller.png)

EEPGD control bit is used to select Program Memory or Data Memory access. If it is set Program Memory is selected and vice versa. Control bits RD, WR are used to initiate read, write, erase operations. These bit cannot be cleared in program, only able to set. These bits are cleared in hardware on the completion of read or write operations.

The WREN control bit is the Write Enable bit, when set it enables write or erase operation. On power-up, the WREN bit will be automatically cleared. The WRERR bit is the Write Error Flag bit, it sets when a write (or erase) operation is interrupted by a MCLR or a WDT Time-out Reset during normal operation. In these situations, we can check the WRERR bit and rewrite the location as the data and address will be unchanged in the EEDATA and EEADR registers.

On the completion of write operation Interrupt flag bit, EEIF in the PIR2 register is set. It must be cleared in program.

EECON2 register is not a physical register and it is used exclusively in the EEPROM write sequence. Reading EECON2 will read all zeros.

Reading from Data EEPROM Memory

1. Write the address of the memory location to be read to EEADR register.
2. To select EEPROM data memory, clear the EEPGD control bit.
3. Set the RD bit to initiate the read cycle.
4. Then we can read data from EEDATA register.

Writing to Data EEPROM Memory

1. Write the address of the memory location to write to EEADR register.
2. Write the 8-bit data to be written in the EEDATA register.
3. To select EEPROM data memory, clear the EEPGD control bit.
4. Set the WREN control bit to enable write operations.
5. Disable Interrupts if enabled in your program. (You may store interrupt register (INTCON) to enable interrupts)
6. Then the special five instruction sequence is executed.
7. Enable Interrupts if using.
8. Disable the program operations by clearing WREN control bit.
9. WR control bit is cleared and EEIF interrupt flag bit is set after completion of the write operation. EEIF bit must be cleared in the program.

MikroC Programming

Functions Developed by Us

MikroC Function to Read Data from Internal EEPROM :

```
unsigned char readEEPROM(unsigned char address)
{
  EEADR = address; //Address to be read
  EECON1.EEPGD = 0;//Selecting EEPROM Data Memory
  EECON1.RD = 1; //Initialise read cycle
  return EEDATA; //Returning data
}
```

MikroC Function to Write Data to Internal EEPROM :

```
void writeEEPROM(unsigned char address, unsigned char datas)
{
  unsigned char INTCON_SAVE;//To save INTCON register value
  EEADR = address; //Address to write
  EEDATA = datas; //Data to write
  EECON1.EEPGD = 0; //Selecting EEPROM Data Memory
  EECON1.WREN = 1; //Enable writing of EEPROM
  INTCON_SAVE=INTCON;//Backup INCON interupt register
  INTCON=0; //Diables the interrupt
  EECON2=0x55; //Required sequence for write to internal EEPROM
  EECON2=0xAA; //Required sequence for write to internal EEPROM
  EECON1.WR = 1; //Initialise write cycle
  INTCON = INTCON_SAVE;//Enables Interrupt
  EECON1.WREN = 0; //To disable write
  while(PIR2.EEIF == 0)//Checking for complition of write operation
  {
    asm nop; //do nothing
  }
  PIR2.EEIF = 0; //Clearing EEIF bit
}
```

To read or write data to EEPROM, you may use built-in MikroC Libraries or user defined functions as following.

Using MikroC EEPROM Libraries

MikroC PRO for PIC Microcontrollers provides library to work with Internal EEPROM. We can easily read/write form EEPROM using the following library functions.

- Eeprom_Read
- Eeprom_Write

Eeprom_Read

Prototype: unsigned short EEPROM_Read(unsigned int address);

Eeprom_Read function reads data from a specified address. Note that parameter address is of integer type, which implies that this functions supports microcontrollers with more than 256 bytes of EEPROM. There must me atleast **20ms delay** between successive using of these routines.

Eeprom_Write

Prototype: void EEPROM_Write(unsigned int address, unsigned short data);

EEPROM_Write function write data to the specified address. As I said above, since the address parameter is of integer type, it supports microcontrollers with more than 256 bytes of EEPROM. There must me atleast **20ms delay** between successive using of these routines. Beware that all Interrupts will be disabled during the execution of this function.

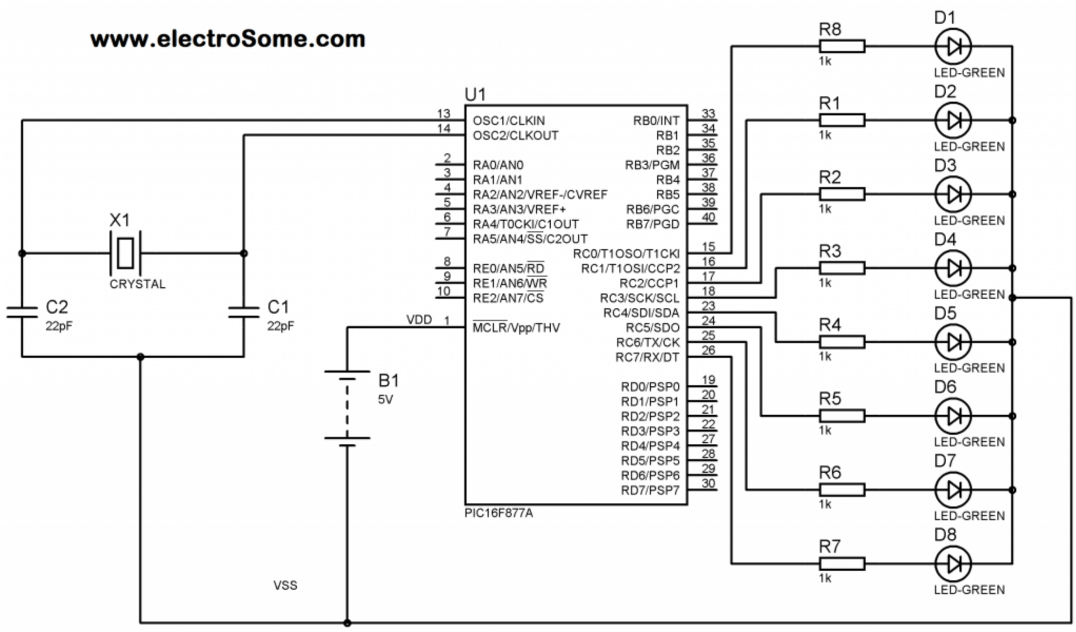
Note: Address ranges from 00h to FFh for devices having 256 bytes while for 128 bytes devices it is 00h to 7Fh.

Mikro Code

```
void main()
{
  unsigned int a, i;
  TRISC = 0;
  do
  {
    for(i=0,a=1;i<8;i++)
    {
      EEPROM_Write(i, a);
      a = a<<1;
    }

    for(i=0;i<8;i++)
    {
      PORTC = EEPROM_Read(i);
      Delay_ms(1000);
    }
  }while(1);
}
```

Circuit Diagram



(https://electrosome.com/wp-content/uploads/2012/06/uieeprompic.png)

Using Internal EEPROM PIC Microcontroller

Note: VDD and VSS of the pic microcontroller is not shown in the circuit diagram. VDD should be connected to +5V and VSS to GND.

In this example we writes 00000001 to the first memory location, 00000010 to second, 000000100 to third etc sequentially up to 10000000. Then it is read sequentially and output through PORTC.

Download Here

You can download the hex file, MikroC source code, Proteus files etc here.

- Using Internal EEPROM PIC Microcontroller (https://electrosome.com/wp-content/uploads/2012/06/Using-Internal-EEPROM-PIC-Microcontroller.zip)

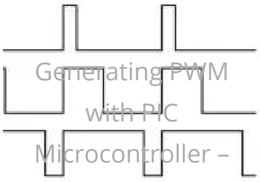
Related Posts:

What is a Microprocessor ?

Using ADC of PIC Microcontroller – MPLAB XC8

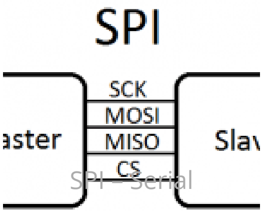
SPI Communication with PIC Microcontroller – MPLAB XC8

I²C Communication with PIC Microcontroller – MPLAB XC8



Generating PWM with PIC Microcontroller – MPLAB XC8

Digital Door Lock using Arduino



SPI – Serial Peripheral Interface

HC-05 Serial Bluetooth Module

Share this post

Author

Ligo George (https://electrosome.com/author/lijoppans/)
Follow Me on LinkedIn (https://www.linkedin.com/in/ligo-george-726b482b/)

32 Comments **electroSome** 1 **Login**

♡ **Recommend** 5 **Share** **Sort by Best**

Join the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

sambasivam • 10 months ago

Hi, how to swap the data from one register to another register without eeprom.

• Reply • Share ›

Emilio Miranda • a year ago

Hi, I have some issues with mikroC EEPROM_Read function. There are some cases when I read a location but no data is saved in ram. I am 100% sure the right addresses are being read since I had written the address after to check if I don´t read any random location. This problem has been solved writing a Delay_ms(20) before the eeprom read instruction. I don´t really know if less time is needed but it works. My question is if someone knows why this problem is solved this way. Thanks.

• Reply • Share ›

Shakil Ahmed • a year ago

Dear sir
Thankyou for your nice blog. I have a question..I want to write a large number I.e 12000000 in EEPROM and want to recall the number for further use. How can I do this?
Thanks

^ | v • Reply • Share ›

Islam Bek • a year ago

HI sir
I downloaded yor project and run the simulation it is giving such warning :
[PIC16 EEPROM] PC=0x0032. Cannot set EECON1<wr> without first setting EECON1<wren> in a previous instruction. [U1]
[PIC16 EEPROM] PC=0x0033. Write to EECON1 sets both EECON1<rd> and EECON1<wr> simultaneously. This is ambiguous - both bits will be considered clear. [U1]

Also,i have tried lot of methods even the mikroc default examples but i am not able to write and read anything to eemprom however when i use eeprom editor it is working, can you give any advice?

^ | v • Reply • Share ›

Ligo George Mod ➦ Islam Bek • a year ago

Are you using the same microcontroller, 16F877A in simulation ?
16F877A and 16F877 are different microcontrollers.

^ | v • Reply • Share ›

Islam Bek ➦ Ligo George • a year ago

oh im sorry i forget to mention, i am usin pic18f452 and mikroc 6.6.3

^ | v • Reply • Share ›

Ligo George Mod ➦ Islam Bek • a year ago

Then you have to make necessary changes in the program to make it compatible with 18F452. Kindly use our forums https://electrosome.com/for... for support as it is outside the scope of above article.

^ | v • Reply • Share ›

Naser • 2 years ago

Dear Ligo,

Your website and tutorials are a big help to the community and I highly appreciate the time and effort you put into this task. God bless you sir.

I have a question please:

How to set and have programmed presets like if I had 16 relays and wanted to save the multiple On or Off states of each/any combination inside the EEPROM memory in different locations for later recall, edit and save again based on the selected memory address?

Thank you again.

Naser

^ | v • Reply • Share ›

Ligo George Mod ➦ Naser • 2 years ago

For, eg: you can use 2 bytes (16 bits) for saving the ON OFF status of 16 relays. You can save multiple status in multiple EEPROM locations and recall or edit it based on different conditions.

^ | v • Reply • Share ›

Naser ➦ Ligo George • 2 years ago

Dear Ligo,

Many thanks for your reply.

I understood your suggestion and tested with a very basic project in mikroC and Proteus, but I have to yet succeed in the part where I can select a memory address from two multiplexed 7-Segments and save the state of each relay as High or Low and then save them all in the selected memory location.

I am using PIC16F877A, briefly with the following setup:

I have 2 buttons named Up and Down which show the memory address/number (total 16 memory locations as 1 to 16) on two multiplexed SSDs, but I am not able to translate them (the SSD number) into a memory location for state save of the relay ports, all on PORTD.

Total 16 relays, as each relay has a button for its On/Off toggle. 8 on PORTB, 8 on PORTC, with state LEDs on their coils for visual status feedback.

All mentioned above are possible to be done using manual memory address pass to the EEPROMWrite(address, data) function, for instance writing the EEPROMWrite(1, PORTB) works fine, but not by getting the SSD numbers and passing it, which should be the case.

Your feedback is highly appreciated in advance.

Naser

^ | ⌄ • Reply • Share ›

Ligo George Mod → Naser • 2 years ago

For eg, let i be the variable on the SSD.
EEPROM address can be 0 to 255. But you need 2 bytes per set... so max sets values of sets will be 0 to 127.
So you can read memory locations 2i & 2i+1

^ | ⌄ • Reply • Share ›

senys → Ligo George • 2 years ago

you only need 2 eeprom addreses to save all 16 relays . one eeprom location can store 8bits of info. For example 00000000 whith could mean that all relays are off and 11111111 could mean that relays are on also this way easy to enable all outputs of a single port in micro just porta = 0b11110000.

^ | ⌄ • Reply • Share ›

Nikos Gorgolis • 2 years ago

Hallo
I have 4 DC motors with one potentiometer each. I read the values of pot from analog pins A0 A1 A2 A3. I want to save 5-6 values of each in and call them by group of four (A0-3) and drive the motors with these values.
How can I do that with eprom?
Any suggestion?

^ | ⌄ • Reply • Share ›

Ligo George Mod → Nikos Gorgolis • 2 years ago

You don't need EEPROM for that... just save it to a variable.

^ | ⌄ • Reply • Share ›

Nikos Gorgolis → Ligo George • 2 years ago

That's the easy way.
My teacher want the hard way...

^ | ⌄ • Reply • Share ›

Oscar ➦ Ligo George • 2 years ago

Hi Ligo,I have a dc motor ." char t " that my "duty cycle=t " I increase it with push button, but how can I save it to a variable that be saved in pic after off and on again?
thank you in advance.

^ | ˅ • Reply • Share ›

Ligo George Mod ➦ Oscar • 2 years ago

It is saving to the internal EEPROM of the PIC. It doesn't need power to keep the data.

^ | ˅ • Reply • Share ›

Richie Ryan S. Tan • 3 years ago

my eeprom always gets restarted, how can I solve that kind of problem?

^ | ˅ • Reply • Share ›

Ligo George Mod ➦ Richie Ryan S. Tan • 3 years ago

EEPROM ? or PIC ?
The above article is for using Internal EEPROM of PIC itself.

^ | ˅ • Reply • Share ›

Yassine Hamroun ➦ Ligo George • 3 years ago

i have the same problem and i use internal EEPROM PIC 16F887

^ | ˅ • Reply • Share ›

Ligo George Mod ➦ Yassine Hamroun • 3 years ago

I don't understand what your problem is. How do you know that PIC's EEPROM gets restarted ?

^ | ˅ • Reply • Share ›

Yassine Hamroun ➦ Ligo George • 3 years ago

my program reads adc channel and if a button is pressed , that value is stored in eeprom . but when i press the button and then cut the power and put it up again the value is lost

^ | ˅ • Reply • Share ›

Ligo George Mod ➦ Yassine Hamroun • 3 years ago

The value will not lost if you store the value correctly. Note that ADC result is 10 bit.. while EEPROM are 8 bits.

^ | ˅ • Reply • Share ›

Leonard Steenkamp • 3 years ago

Hi, It works perfectly!! Thank you. How do I go about saving and recalling unsigned int (16bit)? form EEPROM?

^ | ˅ • Reply • Share ›

Ligo George Mod ➦ Leonard Steenkamp • 3 years ago

You should split integer in to 2 (high significant and low significant 8 bits) and save it...

^ | ˅ • Reply • Share ›

Daniyal • 4 years ago

Thank you so much ! For delivering this EXCELLENT tutorial. Hope you will post some more like that. Many Thanks !

^ | ˅ • Reply • Share ›

Ligo George Mod ➦ Daniyal • 4 years ago

Thanks for the feedback

⌃ ⌄ • Reply • Share ›

yara • 4 years ago

Hi , I'm using EEPROM memory and interrupt as well , what is the final form of the register ????

⌃ ⌄ • Reply • Share ›

Jon Takács • 5 years ago

So, can you read and write to program memory during execution if it is not write protected? So if i get a 32KB MCU and can use that to log sensor data?

⌃ ⌄ • Reply • Share ›

Ligo George Mod ➦ Jon Takács • 5 years ago

Yes... but to log sensor data you don't need to use program flash memory.. use EEPROM data memory... It can be read or write when the device is write protected too..

⌃ ⌄ • Reply • Share ›

Jon Takács ➦ Ligo George • 5 years ago

In interest of lowering the complexity and part count, I want to hit up the highly dense program memory. I need about 16K bytes of data space. I'm having issues finding a MCU under $1 with that density of EEPROM on board... Self Writing seems like the way to go?

⌃ ⌄ • Reply • Share ›

Ligo George Mod ➦ Jon Takács • 5 years ago

If you need more data memory, better use external EEPROM..

⌃ ⌄ • Reply • Share ›

ALSO ON **ELECTROSOME**

Custom Characters on LCD using PIC – MPLAB XC8

6 comments • 3 years ago

Avatar **UnMarciano** — Thank you, you saved my life!

Using ADC of PIC Microcontroller – MPLAB XC8

21 comments • 2 years ago

Avatar **Ligo George** — You can easily convert and use it for 18F4520 if you clearly understand above program.Please let us know if you want any …

Using UART on Raspberry Pi – Python

5 comments • 3 years ago

Avatar **Pierre Marconi** — HII always get a AttributeError: 'module' object has no attribute 'Serial'any suggestions?thanks

Simple Electronic Piano using 555 Timer

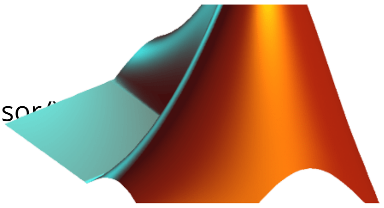
1 comment • 2 years ago

Avatar **Sérgio Toledo** — The problem using the same 1k resistor is that you can not tune the notes, and they are in the wrong pitch.Better will be use …

RELATED **POSTS**

(https://electrosome.com/microprocessor/

(https://electrosome.com/signal-operations-in-matlab/)



(https://electrosome.com/getting-started-with-proteus-beginners-tutorial/)

What is a Microprocessor ?

(https://electrosome.com/microprocessor/)
April 5, 2017 | Ligo George (https://electrosome.com/author/lijoppans/) | 8085 (https://electrosome.com/category/tutorials/8085/), Electronics (https://electrosome.com/category/electronics/), Tutorials (https://electrosome.com/category/tutorials/) | Microprocessor (https://electrosome.com/tag/microprocessor/), processor (https://electrosome.com/tag/processor/) | 0 Comments (https://electrosome.com/microprocessor/#disqus_thread)

Contents1 What is a Microprocessor ?2 Block Diagram of a Computer3 How does a Microprocessor work ?4 Advantages of…

Read More ➞ (https://electrosome.com/microprocessor/)

Signal Operations in MATLAB

(https://electrosome.com/signal-operations-in-matlab/)
May 31, 2013 | Manoj Shenoy (https://electrosome.com/author/manojmshenoy/) | MATLAB (https://electrosome.com/category/tutorials/matlab/), Tutorials (https://electrosome.com/category/tutorials/) | MATLAB (https://electrosome.com/tag/matlab/) | 0 Comments (https://electrosome.com/signal-operations-in-matlab/#disqus_thread)

Contents1 Addition1.1 Functions Used :2 Subtraction3 Multiplication4 Shifting a Signal5 Reversing a Signal6 Linear Convolution of Signals Here I'm going…

Read More ➞ (https://electrosome.com/signal-operations-in-matlab/)

Getting Started with Proteus – Beginners Tutorial

(https://electrosome.com/getting-started-with-proteus-beginners-tutorial/)
May 4, 2013 | Ligo George (https://electrosome.com/author/lij
| Electronics (https://electrosome.com/category/
Tutorials (https://electrosome.com/category/
| Proteus (https://electrosome.com/tag/prote
| 0 Comments (https://electrosome.com/getting-started-with-proteus-beginners-tutorial/#disqus_thread)

Contents1 Switching LED using Push Button Switch1.1 Steps :2 Astable Multivibrator using Transistors Proteus is a design software developed by…

Read More ➞ (https://electrosome.com/getting-started-with-proteus-beginners-tutorial/)