PIC18F4550 On-Chip EEPROM

Introduction to PIC EEPROM

- EEPROM stands for **Electrically Erasable Programmable Read-Only Memory** (EEPROM) is a non-volatile data memory. Data EEPROM is used mainly to store information (data) which is changing frequently and also used for long-term storage of data.
- We can store the data while an application is running. So if there is a power failure (supply off), data will never get lost unlike volatile RAM and we can recollect the data stored in EEPROM.
- It is basically used for the application such as Automatic Electric Meter Reading where the reading needs to be store for continuous calculation and also if a power failure occurs then needs to store current reading also.
- PIC18F4550 has in-built 256 Bytes of data EEPROM. Its address ranges from 0h-FFh.
- The default content of EEPROM data memory is 0xff.

EEPROM has 4 different registers

1. **EEADR**: Register holds the 8-bit address of the EEPROM location which we want to access. It can address up to a maximum of 256 Bytes.
2. **EEDATA**: Register holds the 8-bit data for read/write operation.
3. **EECON1**: It is the EEPROM Control register which needs to be configured for Read/Write operation.
4. **EECON2**: It is not a physical register. It is used in memory erase and write sequence.

EEPROM Registers

Now let's see the EEPROM registers in detail.

EECON1 Register: Data EEPROM Control Register 1

7	6	5	4	3	2	1	0
EEPGD	CFGS		FREE	WRERR	WREN	WR	RD

EECON1 Register

EEPGD: Flash Program or Data EEPROM Memory Select Bit

1= Access flash Program Memory

0= Access Data EEPROM Memory

CFGS: Flash Program/Data EEPROM or Configuration Select Bit

1= Access Configuration Register

0= Access Flash Program/Data EEPROM

FREE: Flash Row Erase Enable Bit:

Used for Flash Memory.

WRERR: Flash Program or Data EEPROM Memory ERROR flag Bit

1= write operation is terminated before completion due to any reset in normal operation or due to improper write attempt.

0= write operation is completed successfully.

WREN: Flash Program or Data EEPROM Write Enable Bit

1= Allow Write cycle to Data EEPROM.

0= Inhibits Write cycle to Data EEPROM.

On power-up, it is cleared to protect against false write or spurious write.

WR: Write Control Bit

1= It initiates Data EEPROM Erase/write operation

0= It indicates EEPROM write cycle is completed.

RD: Read Control Bit

1= Initiates an EEPROM Read

0= It indicates the EEPROM Read cycle is completed.

EECON2

- Before writing data to the EEPROM, a sequence of characters needs to be sent to the EECON2 Register.
- The sequence given below must be written to the EECON2 Register

1. **Write 0x55h**
2. **Write 0xAAh**

- It is recommended to follow the sequence for a write operation.
- **This write initiate sequence and WREN bit together help to prevent an accidental write during power glitches, brown-outs, and software malfunction.**

EEPROM Interrupt

- Also, note that when Write operation is completed **EEIF** interrupt flag is set which indicates write operation is completed.
- This flag is located in PIR2 Register<bit 4>.
- It must be cleared in software.

Note: EEPROM has an endurance of at least 1,000,000 writes/read cycles, so as there is a finite write/read cycle, never read/write EEPROM in a continuous infinite loop.

Steps for EEPROM Programming

Write operation

1. Load the EEADR Register with the address.
2. Load the EEDATA Register with the Data.
3. Configure the EECON1 Register for a write operation by setting EEPGD=0, CFGS=0, and WREN=1.
4. Disable all interrupts.
5. Load a sequence to the EECON2 Register (1st 55h and 2nd AAh) for a write operation.
6. Now start the write operation by setting WR bit to '1' in EECON1 Register.
7. Also, disable global interrupt.
8. Monitor the EEIF flag (PIR2<4>) till it is 0.

```
void EEPROM_Write (int address, char data)
{
    /* Write Operation*/
    EEADR=address;              /* Write address to the EEADR register */
    EEDATA=data;                /* Copy data to the EEDATA register for
                    write to EEPROM location */
    EECON1bits.EEPGD=0;         /* Access data EEPROM memory */
    EECON1bits.CFGS=0;          /* Access flash program or data memory *
    EECON1bits.WREN=1;          /* Allow write to the memory */
    INTCONbits.GIE=0;           /* Disable global interrupt */

    /* Assign below sequence to EECON2 Register is necessary
   to write data to EEPROM memory */

    EECON2=0x55;
    EECON2=0xaa;

    EECON1bits.WR=1;            /* Start writing data to EEPROM memory *
    INTCONbits.GIE=1;           /* Enable interrupt*/

    while(PIR2bits.EEIF==0);    /* Wait for write operation complete */
    PIR2bits.EEIF=0;            /* Reset EEIF for further write operation */
```

Read Operation

1. Load the EEADR Register with the address (e.g. 00h).
2. Configure the EECON1 Register for reading operation by setting

   EEPGD=0, CFGS=0 and WREN=0.
3. Initiates Read operation by setting bit RD=1 in EECON1Register.
4. Read data which is in EEDATA.

```
char EEPROM_Read(int address)
{
    /*Read operation*/
    EEADR=address;        /* Read data at location 0x00*/
    EECON1bits.WREN=0;    /* WREN bit is clear for Read operation*/
    EECON1bits.EEPGD=0;   /* Access data EEPROM memory*/
    EECON1bits.RD=1;      /* To Read data of EEPROM memory set RD=1*/
    return(EEDATA);
}
```

Example

- We are going to use internal Data EEPROM to which first we write data to EEPROM memory location and then read the same data from that memory location. After reading, transmit them serially to the terminal to display.
- Write data to a 1st i.e. 0x00h location and onwards.
- Also, Read data from the same memory location.

PIC18F4550 EEPROM Program

```
}
void EEPROM_WriteString(int address,char *data)
{
    /*Write Operation for String*/
    while(*data!=0)
    {
        EEPROM_Write(address,*data);
        address++;
        *data++;
    }
}

char EEPROM_Read (int address)
{
    /*Read operation*/
    EEADR=address;          /* Read data at location 0x00*/
    EECON1bits.WREN=0;      /* WREN bit is clear for Read operation*/
    EECON1bits.EEPGD=0;     /* Access data EEPROM memory*/
    EECON1bits.RD=1;        /* To Read data of EEPROM memory set RD=1*/
    return(EEDATA);
}
```

Components Used

Powered By MOUSER ELECTRONICS (https://www.mouser.in?utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0)

PIC18f4550
PIC18f4550

X 1

(https://www.mouser.in/ProductDetail/Microchip-Technology/PIC18F4550-I-P?qs=oKK8NaWdAJs8nLDXBGwMXw%3D%3D&utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0)

Datasheet (/components/pic18f4550/1/datasheet)

Components Used

Powered By MOUSER ELECTRONICS (https://www.mouser.in?utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0)

PIC16f877a PIC16f877a	X 1	(https://www.mouser.in/ProductDetail/Microchip-Technology/PIC16F877A-E-P?qs=oKK8NaWdAJsQw%252B2Y417CHw%3D%3D&utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0) Datasheet (/components/pic16f877a/1/datasheet)
PICKit 4 MPLAB PICKit 4 MPLAB	X 1	(https://www.mouser.in/ProductDetail/Microchip-Technology/PG164140?qs=r5DSvlrkXmLKDuYNJImLWw%3D%3D&utm_source=electronicswings&utm_medium=display&utm_campaign=mouser-componentslisting&utm_content=0x0) Datasheet (/components/pickit-4-mplab/1/datasheet)

Downloads

Application_Note	Dow nloa d (/api/download/platform-attachment/24)
PIC18F4550 EEPROM Proteus Simulation	Dow nloa d (/api/download/platform-attachment/25)
PIC18F4550 EEPROM Project File	Dow nloa d (/api/download/platform-attachment/322)

Comments

Comment

Isteward
(/users/Isteward/profile)
2018-05-02 03:55:50

The application will not compile.
error:
mv: cannot stat `build/default/production/USART_communication.d': No such file or directory

Can you please advise on resolving this issue.
Reply Like

lokeshc
(/users/lokeshc/profile)
2018-05-02 04:22:22

I tried at my end and it is compiling and executing properly. Check your ide's and xc8 configuration
Reply Like

Vinivini
(/users/Vinivini/profile)
2019-08-03 09:20:57

Hello

I have tried to associate your work PIC18F4550 ADC with this one, without any success. Do you have idea how it could be possible to read analogic port (ADC works very well) with an write/read process in EEPROM at each measurement. Should be great info. Thanks per advance, with my best regards. Vincent
Reply Like

RobertYahir
(/users/RobertYahir/profile)
2020-12-03 16:11:27

Hi how do I cofigure it to communicate with an external EEPROM?
Reply Like