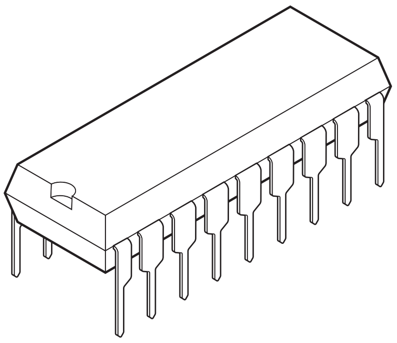
Program PIC16F84A(627)…

Introduction

The Stamp system is ideal for prototyping work as it provides a very quick and convenient way of downloading programs into the Stamp module. However the cost can be restrictive when manufacturing products. The 'Program PIC16F84A…' and 'Program PIC16F627...' options allows an identical Stamp program to be programmed directly into a single microcontroller chip using a conventional programmer. As the PIC16F84A and PIC16F627 are re-programmable devices, the program can be later revised if required.

The 'Program PIC16F84A/627' command programs a PIC16F84A/627 chip with the current BASIC program to mimic a STAMP (NOT the PICAXE system which uses a different pin configuration). However certain restrictions apply due to memory restrictions.



Program Memory

When programming the PIC16F84A/627, the 'interpreter' and the 'tokenised program' are both programmed into the microcontroller. The original Stamp system uses a PICmicro interpreter with 1k internal memory, with a 256 byte external memory chip for the tokenised program. As the PIC16F84A/627 only has 1k internal memory, it is not possible to fit both the interpreter and tokens into the PIC16F84A/627 memory!

Therefore the interpreter memory intensive commands serin, button and debug have been removed from this function. If your program does not contain these commands it will function correctly.

If your program does make use of these commands it is necessary to use an alternative approach. Convert the BASIC program into assembler code, and then program this assembler code into the PIC16F84A/627. As this approach uses a completely different process it is more likely to fit into the microcontroller.

Data Memory

The PIC16F84A/627 has an additional 64 bytes of data memory. Therefore the commands eeprom, read and write are mapped to this additional memory space. Care must be taken to ensure that only the values 0 to 63 are used as addresses with these three commands.

PORTA Pins

The Stamp system uses the microcontroller porta pins for the external EEPROM and computer connections. These connections are not required for the PIC16F84A/627, freeing five additional pins (porta,0 to porta,4). To make these pins available to the PIC16F84A user two new variables PINSA and DIRSA are defined. These two variables are mapped on top of B12 and B13, which are not required for the gosub stack (as with the original Basic Stamp system). Therefore, as with the original Basic Stamp, B12 and 13 (and W6) should not be used as general purpose registers.

PORTA Control Commands

The following commands operate on PORTA in an identical manner to the way the standard commands operate on PORTB.

```
higha     ‘ switch pin (0-4) high
lowa      ‘ switch pin (0-4) low
inputa    ‘ make pin (0-4) an input
outputa   ‘ make pin (0-4) an output
```

Due to programming limitations there is no PORTA equivalent of the reverse or toggle commands. In addition the PINSA and DIRSA variables may be addressed directly.

To set the five available porta pins as outputs:

```
let dirsa = %00011111
```

To switch pin porta,0 high:

```
let pinsa = pinsa OR %00000001
```

To switch pin porta,0 low:

```
let pinsa = pinsa AND %11111110
```

To toggle pin porta,0:

```
let pinsa = pinsa XOR %00000001
```