In-circuit programming of the Philips 87C576 microcontroller

AN455

Author: Bill Houghton

Introduction

The 87C576 includes *two* separate methods of programming the EPROM array, the traditional modified Quick-Pulse method, and a new On-Board Programming technique (OBP).

Quick Pulse programming is a method using a number of device pins in parallel (see Figure 1) and is the traditional way in which 87C51 family members have been programmed. The Quick-Pulse method supports the following programming functions:

– program USER EPROM
– verify USER EPROM
– program KEY EPROM
– program security bits
– verify security bits
– read signature bytes

The Quick-Pulse method is quite easily suited to standard programming equipment as evidenced by the numerous vendors of 87C51 compatible programmers on the market today. One disadvantage is that this method is not well suited to programming in the embedded application because of the large number of signal lines that must be isolated from the application. In addition, parallel signals from a programmer would need to be cabled to the application's circuit board, or the application circuit board would need to have logic built-in to perform the programming functions. These requirements have generally made in-circuit programming using the modified Quick Pulse method impractical in almost all 87C51 family applications.

On-Board Programming (OBP)

The Philips On-Board Programming (OBP) facility has made in-circuit programming in an embedded application possible with a minimum of additional expense in components and circuit board area. The OBP function uses three pins in addition to V_{CC} and V_{SS} (see Figure 2). The diode shown is added so that in the absence of the five pin connector the V_{PP} pin will be pulled up above the V_{IH} level in normal system operation allowing the 87C576 to execute from internal code memory.

The On-Board Programming facility consists of a series of internal hardware resources coupled with internal firmware to facilitate remote programming of the 87C576 through the serial port. This firmware is provided by Philips and embedded within each 87C576 device. The On-Board Programming facility supports the following programming functions:

– program USER EPROM
– display USER EPROM
– program security bits
– verify security bits
– specify timing parameters

The OBP function is invoked by having the EA/V_{PP} pin at the V_{PP} voltage level at the time that the part exits reset. The OBP function uses five pins (TxD, RxD, V_{SS}, V_{CC}, and V_{PP}). Only a small connector needs to be available to interface your application to an external circuit in order to use this feature. The V_{PP} supply should be adequately decoupled and V_{PP} not allowed to exceed datasheet limits.

Using the On-Board Programming (OBP)

The OBP feature allows for a wide range of baud rates to be used in your application, independent of the oscillator frequency. It is also adaptable to a wide range of oscillator frequencies. This is accomplished by measuring the bit-time of a single bit in a received character. This information is then used to program the baud rate in terms of timer counts based on the oscillator frequency. The OBP feature requires that an initial character (an uppercase U) be sent to the 87C576 to establish the baud rate. The OBP firmware provides auto-echo of received characters.

Once baud rate initialization has been performed, the OBP firmware will only accept Intel Hex-type records. Intel Hex records consist of ASCII characters used to represent hexadecimal values and are summarized below:

:NNAAAARRDD..DDCC<crlf>

In the Intel Hex record, the "NN" represents the number of data bytes in the record. The "AAAA" string represents the address of the first byte in the record. If there are zero bytes in the record this field is often set to 0000. The "RR" string indicates the record type. A record type of "00" is a data record. A record type of "01" indicates the end-of-file mark (Intel Hex records were once used to load microprocessor programs from a paper tape, hence the need for an end-of-file marker). In this application additional record types will be added to indicate either commands or data for the OBP facility. The maximum number of data bytes in a record is limited to 16 (decimal). OBP commands are summarized in Table 1.

As a record is received by the 87C576 the information in the record is stored internally and a checksum calculation is performed. The operation indicated by the record type is not performed until the entire record has been received. Should an error occur in the checksum, the 87C576 will send an "X" out the serial port indicating a checksum error. If the checksum calculation is found to match the checksum in the record then the command will be executed. In most cases successful reception of the record will be indicated by transmitting a "." character out the serial port (displaying the contents of the internal program memory is an exception).

In the case of a Data Record (record type 00) an additional check is made. A "." character will NOT be sent unless the record checksum matched the calculated checksum and all of the bytes in the record were successfully programmed. For a data record an "X" indicates that the checksum failed to match and an "R" character indicates that one of the bytes did not properly program. It is necessary to send a type 02 record (specify timing parameters) to the 87C576 before programming data or security bits.

Programming requires two delay times for the 87C576. A 50µS programming pulse needs to be generated along with a 10µS minimum delay between pulses. The OBP facility was designed so that specific crystal frequencies were not required in order to generate baud rates or time the programming pulses. The user thus needs to provide the 87C576 with information required to generate the proper timing. Record type 02 is provided for this purpose. This function uses timer 0 of the 87C576. Thus the times are specified in terms of the timer 0 reload values necessary to produce the required time periods based on the oscillator frequency in the application.

50µS timer value = –FOSC * 50/12
10µS timer value = –FOSC * 10/12

The two security bits can be programmed using the 03 record type. The record length is one data byte. Bit 0 corresponds to security bit 0 and bit 1 corresponds to security bit 1. A logical one in either bit position indicates that the particular security bit is to be programmed. Either zero, one, or two bits may be specified.

Table 1.

RECORD TYPE	COMMAND/DATA FUNCTION
00	Indicates a data record. Programs the part with data indicated in the record starting with load address in the record.
01	EOF record, no operation
02	Specify timing parameters. The record length is 3 bytes. load address = xxxx 1st byte = timer count for 50uS programming pulse 2nd byte = timer count for 10uS delay between pulses 3rd byte = 0AH
03	Program security bits The record length is 1 byte. load address = xxxx 1st byte = sec bit values (xxxx xxB2B1)
04	Display contents of USER EPROM array The record length is zero (no data bytes are in this record). load address =xxxx
05	Verify security bit status The record length is zero (no data bytes are in this record). load address =xxxx

Record type 04 causes the contents of the entire 8Kbyte EPROM array to be sent out the serial port in a formatted display. This display consists of an address and the contents of 16 bytes starting with that address. No display of the device contents will occur if security bit 2 has been programmed. Instead an “S” character will be sent to indicate a security violation. The dumping of the device data to the serial port is terminated by the reception of any character.

The status of security bits can be verified using a record type of 05. Two characters are sent out the serial port representing a byte quantity. Bits 1 and 2 of this byte represent security bits 1 & 2, respectively, where a logical one means the security bit has not been programmed.

Some examples of Intel–Hex records used with On–Board Programming are shown below:

```
:03000002CEF60A2D  <set timing parameters for 12MHz osc>
:03000000010203F7  <program data = 01,02,03>
:0100000301FB      <program security bits 1>
:00000004FC        <display contents of EPROM>
```

Using Your PC as an OBP Programmer

While the process of using OBP is relatively simple, a PC can be used to simplify the procedure. The PC can calculate the timing parameters based on your oscillator frequency, perform the baud rate initialization, perform file downloading/programming, and translate the EPROM contents display into a blank-check feature. At the time of this writing, a DOS–based version of such a program is available from the Philips Microcontroller Bulletin Board (1-800-451-6644). An OBP demo board for the 87C576 is available for qualified customers through your local Philips sales office or representative.

In-circuit programming of the Philips 87C576 microcontroller

AN455

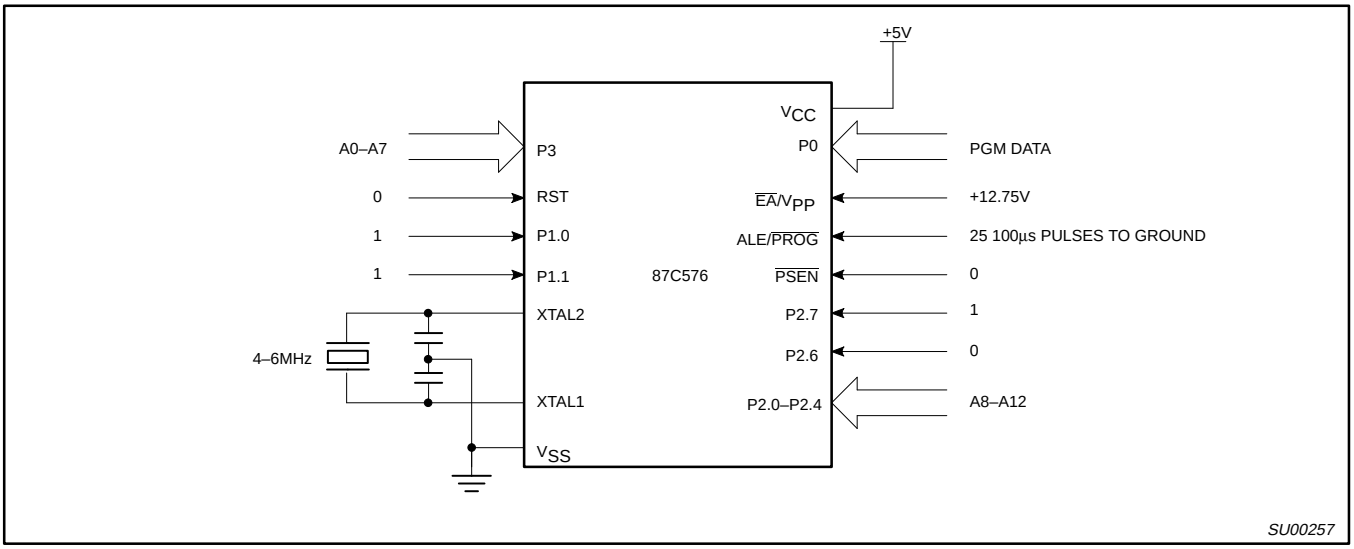

Figure 1. "Quick-Pulse" Programming Using Parallel Pins

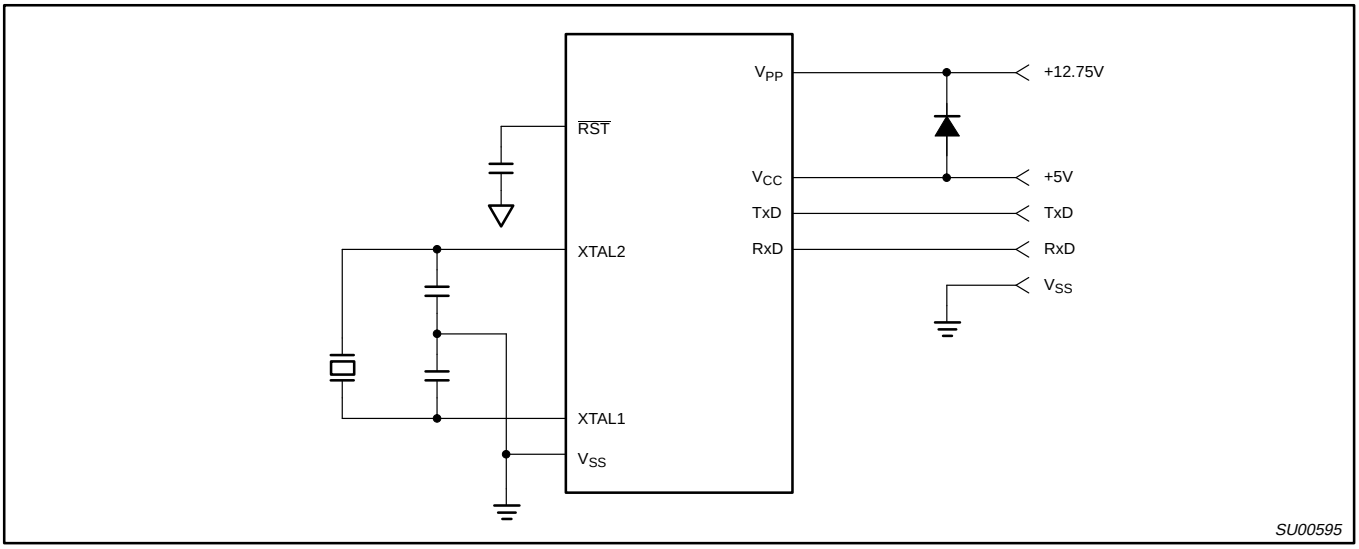

Figure 2. On-Board Programming Uses a Minimum of Pins

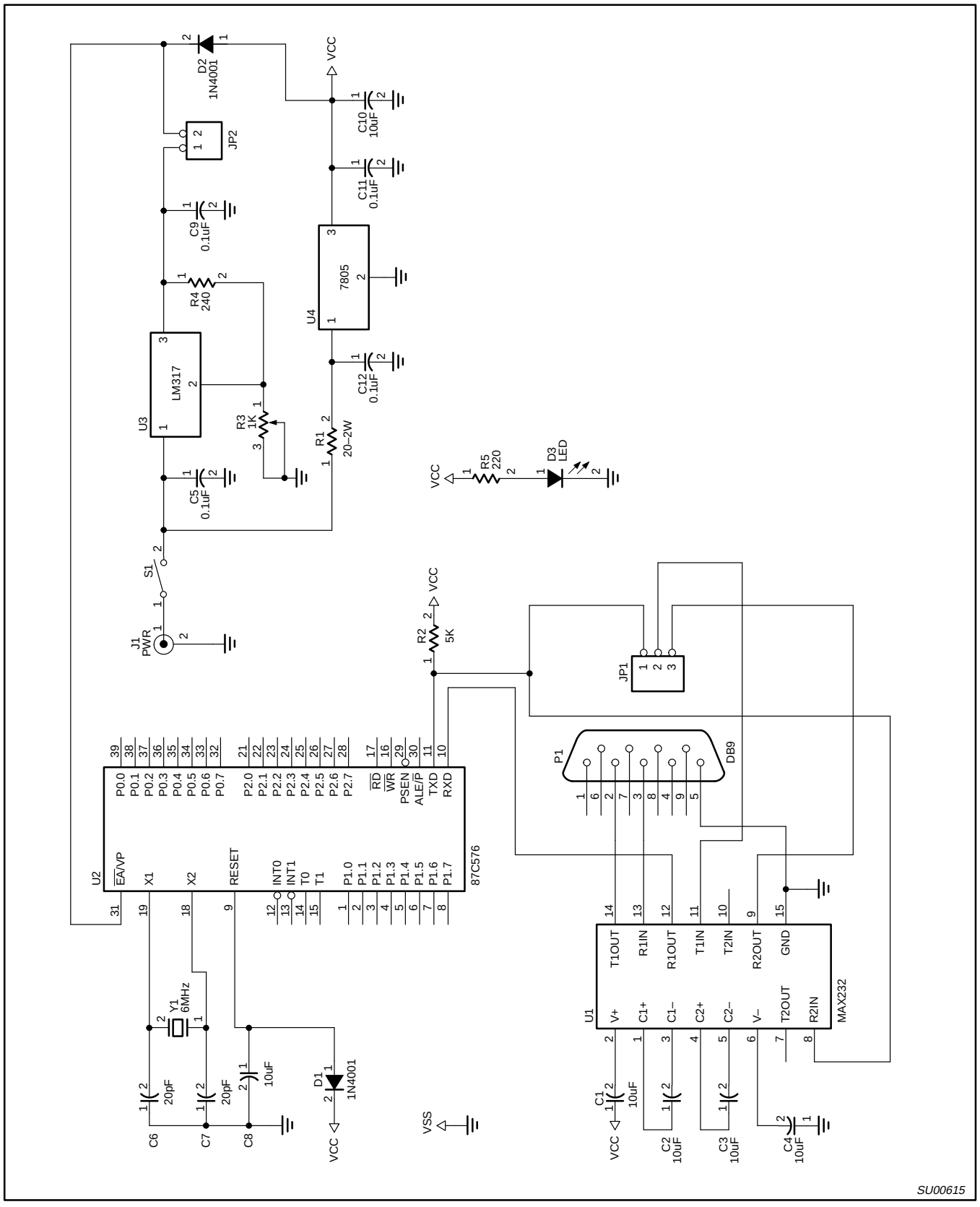

Figure 3. Schematic of the OBP Demoboard for the 87C576

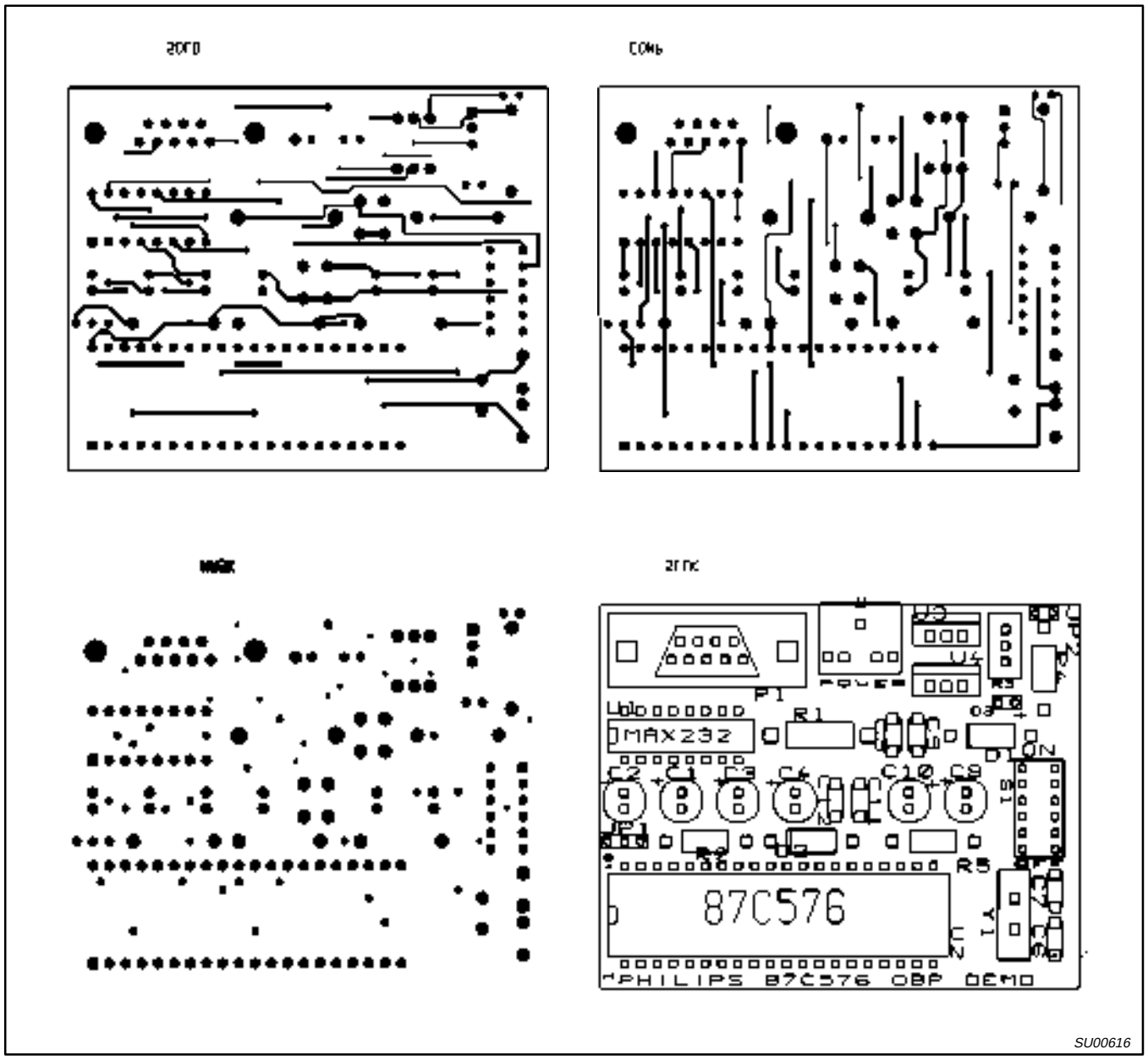

Figure 4.