MCT Programmer Support

COP8SAx7 Family Programming Specification and Application Notes for In Circuit Programming (ICP) and Personal Programmer Design

HISTORY:
Rev 1.0, 10/23/97 - First draft
Rev 2.0, 11/18/97 - Second draft; Reading signatures dropped; Added HV pins;
Rev 3.0, 2/1/98 - Release; Corrected tpd1; Added max fcki; Completed timing and flow diagrams; Added notes;

If you have comments or suggestions concerning this document, please contact:

Dave Katra, MCT Tools
dave.katra@nsc.com
fax 408-721-6239

TABLE OF CONTENTS

1.0 INTRODUCTION

This document provides all the details for reading and programming COP8Sxx family devices in an In-Circuit or Personal Programmer environment. This information is intended for the use of end users who want to implement end-of-line production programming, board-level programming for other purposes such as field upgrades or maintenance, or simply build a personal programmer.

External Programming Mode details and algorithm flow charts are covered, along with details of reading and writing data to the internal EPROM and Configuration Array. Additional information is included to assist in design and implementation of ICP Systems, or a Personal Programmer. A separate COP8 databook or datasheet must be referred to for functional operation details.

For commercial device programmer manufacturers, a separate programmer specification document is available which includes additional details about device signatures, data manipulation, and production programming equipment operation.

2.0 DEVICE OVERVIEW

The COP8Sxx Series is a family of single chip CMOS microcontrollers with on-chip EPROM as program memory. The programmable elements are accessed in a special mode which bypasses all of the device's in-system functional elements, and directly reads and writes to the EPROM using conventional EPROM memory programmer techniques. A programmable configuration register enables internal features, and 8 programmable user bytes are provided for end-user use.

Devices are shipped in a variety of EPROM and internal RAM configurations, and in a variety of packages and pinouts.

2.1 SUPPORTED DEVICES

Before National will support devices for ICP or Personal Programmer designs, each product must be proven (by programming history), to be easy to program with consistently high programming yields. Additionally, the programming algorithm must be stable with tested documentation.

The following family devices are covered by this document, and are currently qualified for in-circuit, and end-user programming:

COP8SAA7-16pin	(1k)(DIP; SOIC)
COP8SAA/AB/AC7-20pin	(1k/2k/4k)(DIP; SOIC)
COP8SAA/AB/AC7-28pin	(1k/2k/4k)(DIP; SOIC)
COP8SAC7-40pin	(4k)(DIP)
COP8SAC7-44pin	(4k)(PLCC)

3.0 PROGRAMMING MODE

External programming mode is entered when high voltage is applied to the G6 and VPP pins. The pin functions of the device become reconfigured for programming and reading of the EPROM and Configuration array, and are different from the datasheet functions. Programming is done in a serial manner by clocking in address and data on two pins, then applying a program pulse. A special Compare output is provided for sensing correctly programmed bytes. Data is also read out serially for normal read and verify operations.

3.1 DEVICE ARCHITECTURE

Once in the programming mode the EPROM, configuration register, and user bytes can be read and programmed at the following addresses. The device physical address may be different from the programmer RAM address:

Device Address	RAM Address	Allocation
0000-MaxAdd	0000-MaxAdd	1k-32k of EPROM for customer pattern, depending on device. MaxAdd=Last valid EPROM address. (Read and Write; Securable)
20	MaxAdd +1	ECON Register - 1 byte; Auto configured by HEX file and master; **Includes Security Bit**; (Read and Write; Write secured only)
00-07	MaxAdd +2-9	User Bytes - 8 bytes for end-user use. (Read and Write; Not Secured) Note: The User Bytes can be ignored for typical in-circuit applications

3.2 ADDRESSING MODES

The EPROM and Configuration arrays are addressed in two modes:

Sequential Mode
For the EPROM (Addresses less than or equal to MaxAdd), Sequential Mode is used, always starting at address 0000. The programmer supplies only the input data, and the COP8 device provides the EPROM address internally. The address is automatically incremented after each serial byte of data.

Random Mode
For the ECON and User Bytes (Addresses greater than MaxAdd) Random Mode is used. The programmer supplies an external address, along with the input data.

3.3 PROGRAMMING MODE INTERFACE

10-20 external pins (Dependent on device pin count, and PCB design), including power and ground, are addressed in the Programming Mode. Most of these pins must be isolated from the PCB's normal circuit connections before attempting in-circuit programming or reading of the device.

continued on next page.....

External Programming Mode device pin functions. (Different from datasheet pin functions!)

PIN **(note*)**	**I/O** **(note**)**	**LEVELS**	**ICP ISOLATE**	**FUNCTION (note)**
VCC (Vcc)	input	0-6.5v	yes	Device Vcc. Varies by operation.
G6 (G6)	input	0-13v	yes	High volts reconfigures device for program and read.
VPP (RST)	input	0-13v	yes	High volts for programming and reading circuits.
GND (GND)	input	0v	preferred	Device ground. May be left connected to PCB if programming equipment is well grounded to PCB.
SEL (G4)	input	Vil / Vih	yes	Vil selects Sequential Mode. Vih selects Random Mode.
PCADD(G1)	input	Vil / Vih	yes	Serial address input in Random Mode only.
DIN (G7)	input	Vil / Vih	yes	Serial data input.
DOUT (G0)	output	Vol/ Voh	yes	Serial data output, and Compare output
/CKI (CKI)	input	Vil / Vih	yes	Address and Data clock. Negative going pulse.
PGM (G2)	input	Vil / Vih	yes	Pulse to Vih for program pulse.
NOE (G3)	input	Vil / Vih	yes	Vil enables DOUT; Vih tristates DOUT.
NC (F,L,C)	output	HIZ	no	Unused pins; Driven to HIZ by device. May be left connected to the PCB circuits
HV (D0-D7, G5)	output	0-6.1v	optional	Unused pins; Driven to Vcc-.25v by device. PCB connections must be able to accept 6.0+v levels.

*Note: (xxx) indicates the normal COP8 datasheet function for reference.

**Note: I/O indicates pin is driven by device (output) or programming circuits (input) while G6 and Vpp are above Vih levels. When G6 and Vpp are at 0v levels, the device is in its datasheet mode, and all pins are at their normal RESET states.

3.4 DEVICE PINOUTS (Refer diagrams at end of document)

3.5 AC AND DC SPECIFICATIONS

Symbol	**Parameters**	**Min**	**Typ**	**Max**	**Conditions**
Vccp	PROG/COMPARE Vcc	6.20v	6.25v	6.50v	VPP, G6=Vpp
Vccv	VERIFY Vcc	5.70v	5.80v	5.90v	VPP, G6 =Vpp
Vccr	READ Vcc	2.70v	4.20v	5.50v	VPP, G6 =Vpp
Icc	Icc read and program			25ma	VPP, G6 =Vpp
TrVcc	Vcc rise time	10ns		50us	On power up
Vpp	Program Mode Voltage	12.50v	12.75v	13.00v	VPP & G6 pins
Ipp	Vpp current on VPP			50ma	VPP/RST = Vpp
Continued on next page					

Symbol	Parameters	Min	Typ	Max	Conditions
IppG6	Vpp current on G6			200ua	G6 pin = Vpp
Tr Vpp/G6	Vpp/G6 rise time	1us		100us	1-12v
Vih	High input levels	4.00v		Vcc	Vccp and Vccv
Vih	High input levels	.7xVcc		Vcc	Vccr
Vil	Low input levels	GND		.2xVcc	
td1	VCC to G6	Min of 15us or 5xTrvcc; Whichever is greater.			
td2,	G6 to VPP	100ns			
td3	VPP to /CKI high	15us			
td4	SEL to /CKI high	5ns			
T1	/CKI Vil time between bytes	100ns			
td6	/CKI low to prog pulse	100ns			
td7, td8	tpw3 to tpw3 to /CKI high	2.6us			
ts1	DIN or PCADD to /CKI	0ns			
th1	/CKI to DIN or PCADD	25ns			
ts2, th2	DIN to G6; RST to DIN	20ns		7 /CKIs	
tpw1, tpw2	/CKI high / low time	30ns			Vcc = 4.0v-6.5v
tpw1, tpw2	/CKI high / low time	60ns			Vcc = 2.7v-4.0v
fcki	/CKI frequency			15MHz	Vcc=4.5-6.5v
fcki	/CKI frequency			7.5MHz	Vcc=2.7v
tpd1	/CKI falling to DOUT valid			250ns	Voh=2v; Vcc=2.7
tpd2	Prog pulse to Compare	2.5us			PROGRAM only
tpw3	Program pulse	50us	50us	55us	Up to 6 attempts
tpw4	Secure program pulse	300us	300us	310us	1 attempt only
#Tpw	# Program Pulses			6	Except Security=1

4.0 READING AND PROGRAMMING DATA (Refer to timing diagrams)

IMPORTANT: For in-circuit programming, the COP8 device pins MUST be properly isolated before putting it into the programming mode. How this is actually done is customer PCB design specific, and may either be an automatic process or a manual process. Refer to sections 6.0, 7.0, and 8.0 for more information on ICP design considerations.

4.1 ENTERING AND EXITING THE PROGRAMMING MODE

After isolation, a specific sequence of events must occur each time the device enters, or exits the programming mode. Specific attention must be paid to the Vcc/Vpp/G6 rise times, the period of time when Vcc is on the device while Vpp/G6 are less than 6v, and the DOUT / HV pins. After putting voltages on the device the internal circuits must be initialized before reading or programming data.

ENTER and INITIALIZE:

1. Set all pins low. (Device is in normal Reset).
2. DOUT and HV pins to HIZ. (At all times)
3. VCC=Vcc. Delay td1. (Check rise time)
4. DIN, NOE to high. Delay ts2.
5. G6=Vpp. Delay td2. (Check rise time)
6. VPP=Vpp. Delay th2. (Check rise time)
7. NOE, DIN to low. Delay td3.
8. (Initialize): /CKI to high.
9. Pulse /CKI 10 times to low.
10. /CKI to low (Proceed to program/verify/read)

EXIT:

1. DOUT and HV pins to HiZ. (Unchanged)
2. DIN and NOE to High. Delay td2
3. VPP=0v. Delay td2
4. G6=0v. Delay td2.
5. VCC=0v.

4.2 RANDOM MODE - ECON AND USER BYTES

The ECON and User Bytes are always addressed in the Random Mode. The sequencing, waveforms, and voltages are identical for READ, and VERIFY, but during PROGRAM there is a slight difference in how the programmed byte is verified. The Random Mode may be entered at any address.

The Random Mode is enabled when the SEL pin is at Vih. Address (nn) is clocked in on PCADD by the programmer at the same time as DIN.

4.2.1 Random Mode - READ, VERIFY

After setting SEL to Vih, address "nn" is clocked in on PCADD (Note that 9 /CKI pulses are used to input 8 address bits; No valid data is output yet). On the next series of 9 /CKI pulses the next desired address is clocked in, and the ECON data byte or User Byte at address "nn" is output serially on DOUT. This 9 /CKI pulse cycle is repeated for as many addresses as necessary.

1. DOUT to HIZ. DIN to low. NOE to low.
2. SEL to high. Delay td4.
3. /CKI to high.
4. (Clock in address): Bit 0 of address "nn" to PCADD. Delay ts1.
5. continued……

6. pulse /CKI. Delay th1.
7. Repeat steps 3 and 4 for address bits 1-7.
8. pulse /CKI. (Note that 9 /CKI pulses are used to input 8 address bits)
9. /CKI to low. Delay T1.
10. (Read out data from address nn): /CKI to high.
11. (Set up optional next-address bits before each /CKI pulse) Pulse /CKI. Pulse /CKI.
12. Sense DOUT for bit 0 (1=Programmed bit; 0=Erased bit).
13. Pulse /CKI
14. Repeat steps 11-12 for data bits 1-7. (Note that 9 /CKI pulses are used to ouput 8 data bits)
15. /CKI to low.

4.2.2 Random Mode - PROGRAM / COMPARE

After setting SEL to Vih, a data byte and address "nn" are clocked in on DIN and PCADD (Note that 9 /CKI pulses are used to input 8 data/address bits). A program pulse is applied, and the Compare bit is sensed on DOUT (This Compare bit is a comparison of the input data byte and the ECON byte or User Byte output data). Program pulses are applied until DOUT goes to Vih, then a single over-program pulse is applied. If the maximum attempts are reached without a Compare, then the device is rejected. This cycle is repeated for as many addresses as necessary.

1. DOUT to HIZ. DIN to low. NOE to low.
2. SEL to high. Delay td4.
3. /CKI to high.
4. (Clock in data and address): Bit 0 of data to DIN; Bit 0 of address "nn" to PCADD. Delay ts1.
5. pulse /CKI. Delay th1.
6. Repeat steps 4 and 5 for data and address bits 1-7.
7. pulse /CKI. (Note that 9 /CKI pulses are used to input 8 data/address bits)
8. /CKI to low. Delay td6.
9. Pulse PGM tpw3. Delay tpd2.
10. Sense Compare on DOUT: Voh=Programmed byte; Vol=Pulse PGM again.
11. Repeat steps 9-10 until DOUT=Voh, or the max # of pulses have been applied.
12. If DOUT=Voh, apply 1 over-program pulse on PGM. If max # pulses then reject the device.
13. Continue to next address at step 3, or reject part.

4.3 SEQUENTIAL MODE - EPROM

The EPROM is ALWAYS addressed in the Sequential Mode. The sequencing, waveforms, and voltages are identical for READ, BLANK CHECK, and VERIFY, but during PROGRAM there is a slight difference in how a programmed byte is verified.

The Sequential Mode is enabled when the SEL pin is set to Vil. Addressing ALWAYS begins at 0000. The full address is provided internally by the device, and the programmer only provides programming data input. The internal address increments after every series of input clocks on /CKI.

4.3.1 Sequential Mode - READ, BLANK CHECK, VERIFY

AFTER initialization, 9 /CKI pulses are input to enable address 0000 (no valid data is output yet). On the next series of 9/CKI pulses the data byte at address 0000 is output serially on DOUT (Note that 9

/CLK pulses are used to output 8 bits of data), and the internal address is automatically incremented. This 9 /CKI pulse cycle is repeated up to the maximum EPROM address (MaxAdd).

1. DOUT to HIZ. DIN to low. NOE to low.
2. SEL to low. Delay td4.
3. /CKI to high.
4. (Enable address 0000): Pulse /CKI 9 times.
5. /CKI to low. Delay T1.
6. (Read out data from address 0000): /CKI to high.
7. Pulse /CKI. Pulse /CKI. Delay tpd1.
8. Sense DOUT for bit 0 (1=Programmed bit; 0=Erased bit).
9. Pulse /CKI
10. Repeat steps 8-9 for data bits 1-7. (Note that 9 /CKI pulses are used to ouput 8 data bits)
11. /CKI to low. Delay T1. (Address increments to 0001)
12. Repeat steps 6-11 for ALL addresses 0001-MaxAdd.

4.3.2 Sequential Mode - PROGRAM / COMPARE

AFTER initialization the data byte for address 0000 is clocked in on DIN. (Note that 9 /CKI pulses are used to input 8 data/address bits). A program pulse is applied, and the Compare bit is sensed on DOUT (This Compare bit is a comparison of the input data byte and the EPROM data byte for address 0000). Program pulses are applied until DOUT goes to Voh, then a single over-program pulse is given. If the maximum attempts are reached without Compare, then reject the device. The internal address is automatically incremented, and this cycle is repeated at each address up to the maximum EPROM address (MaxAdd).

1. DOUT to HIZ. DIN to low. NOE to low.
2. SEL to low. Delay td4.
3. /CKI to high.
4. (Input data byte for address 0000): Bit 0 of data to DIN. Delay ts1.
5. pulse /CKI. Delay th1.
6. Repeat steps 4 and 5 for data bits 1-7.
7. pulse /CKI. (Note that 9 /CKI pulses are used to input 8 data bits)
8. /CKI to low. Delay td6.
9. Pulse PGM tpw3. Delay tpd2.
10. Sense Compare on DOUT: Voh=Programmed byte; Vol=Pulse PGM again.
11. Repeat steps 9-10 until DOUT=Voh, or max # of pulses have been applied.
12. If DOUT=Voh, apply 1 over-program pulse on PGM. If max # pulses then reject the device.
13. Continue to next address at step 3, or reject part.

4.4 THE SECURITY BIT

The Security bit is read and programmed in the Random Mode at address 20 (The same as ECON; Section 4.2.1 and 4.2.2, but with several differences):

1. The Security bit is always programmed seperately from the other ECON data. Program data byte 20h into the location.
2. Only a single program pulse of tpw4 is applied. No overprogram pulse is needed

3. The Security bit is read/verified by reading the ECON data byte out serially on DOUT. (Section 5.5 has more details).
4. If bit 5 is a zero, the device has failed to secure, and is rejected. If a one, then the device is secured.
5. The verify of the security bit after program may be skipped completely, because of the guaranteed design of the device. (Although the user should confirm that they are properly executing the secure operation).

5.0 PROGRAMMER OPERATIONS

5.1 DEVICE SIGNATURES

Implementing device signature recognition is not necessary for end-users, or in-circuit programming. Programmer memory size is set by the menu selection, and correct device type on the PCB is assumed to be correct. Signature checking also adds significant complexity to the programming algorithms.

NOTE: <u>For in-circuit programming, signature checking may be used to make sure the COP8 device on the PCB is the correct device type.</u> Contact MCT Tools support for details if needed.

5.1.1 Signature Alternatives

The programming equipment sets the memory size and device configuration according to the the device menu selection. The table below gives the menu selection, and memory size for each supported device type:

Menu Selection	EPROM Size	Max Add in RAM (hex)	ECON RAM address (hex)	User Bytes RAM address (hex)
COP8SAA716	1k EPROM	3FF	400	401-408
COP8SAA720	1k EPROM	3FF	400	401-408
COP8SAA728	1k EPROM	3FF	400	401-408
COP8SAB720	2k EPROM	7FF	800	801-808
COP8SAB728	2k EPROM	7FF	800	801-808
COP8SAC720	4k EPROM	FFF	1000	1001-1008
COP8SAC728	4k EPROM	FFF	1000	1001-1008
COP8SAC740	4k EPROM	FFF	1000	1001-1008
COP8SAC744	4k EPROM	FFF	1000	1001-1008

5.2 THE EPROM

The EPROM is 8 bits wide, and is ALWAYS read and programmed in the Sequential Mode, and always starting at device address 0000 (Programmer RAM address 0000 maps to device address 0000). The size of the EPROM is determined by the programmer menu selection.

The EPROM data in programmer RAM should always clear to 00 (erased) before reading a master device, or downloading a HEX file into the programmer.

RAM and Device address 0000-MaxAdd

5.3 ECON REGISTER

The ECON register is 8 bits wide, and is ALWAYS read and programmed in the Random Mode at address 20 in the device. **The ECON bits in programmer RAM are located at address MaxAdd+1,*

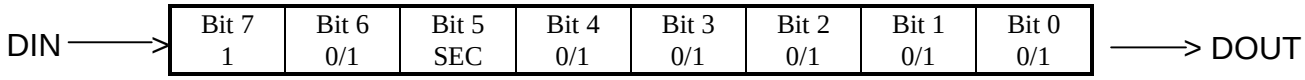
Continued……

and always included in blank check, verify, checksum, and an output HEX file.

The ECON data location in programmer RAM should always be cleared to 00 (erased along with the rest of the programmer RAM) before reading a master device, or downloading a HEX file.

*Bits 7 and 5 have special treatment (See notes below):

Before programming the ECON, the data from RAM must be OR'd with 0x80, then AND'd with 0xDF. This insures that the Compare function works correctly (Always programming a 1 in bit 7), and insures that the Security bit 5 is NEVER programmed at the same time as the other data bits. The programmer RAM data is left unchanged.



RAM address MaxAdd+1 = Device address 20

*Bits 0-4,6 are configured by a HEX file, master device, or manual editing.

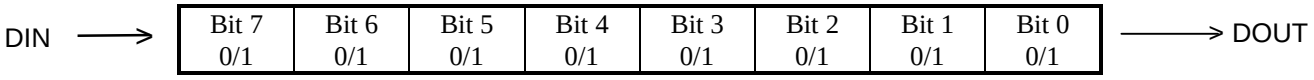
*Bit 5 (Security bit) may be configured by a HEX file, but normally should default to 0. (Turning "ON" the secure function at the programmer menu should NOT change this bit in programmer RAM) . *Bit 5 should **NEVER** be programmed as a 1 when programming the ECON register data. It is always programmed seperately during the secure operation (Section 5.5)*

*Bit 7 is a test bit, and may actually be a 0 or 1 in the device as it is shipped from the factory. ***It is ALWAYS ignored in Read, Blank Check, and Verify, and is ALWAYS set to 0 in programmer RAM to assure consistent programmer checksums.*** (But, it is ALWAYS programmed to a 1 in the device to insure that the device COMPARE function works correctly).

5.4 USER BYTES (Optional: Can be skipped if not used)

The User Bytes consist of 8 bytes (64 bits) of user-defined data, and is ALWAYS read and programmed in the Random Mode at device addresses 00-07. **The USER BYTES in programmer RAM are located at addresses MaxAdd+2-9, and always included in blank check, verify, checksum, and the output HEX file .*

*The USER BYTES in programmer RAM should clear to 00 (erased) before reading a device, or downloading a file.
*The USER BYTES are not securable, and can always be read.



Ram addresses MaxAdd+2-9 = device addresses 00-07

*The USER BYTES are configured by the HEX file, master device, or programmer RAM editing.

5.5 SECURITY BIT

The security bit consists of a single bit in the ECON register (bit 5; SEC), and is ALWAYS read and programmed in the Random Mode at device address 20 (Exactly the same as ECON).

*The security bit may be configured in the programmer RAM by a HEX file, but normally should default to 0. Turning "ON" the secure function at the programmer menu should NOT change this bit in programmer RAM, and the checksum, which includes the ECON, should remain unchanged.

*After programming the EPROM and ECON, the programmer should automatically secure the device if the secure function is enabled from the programmer menu.

The security bit should **NEVER be programmed as a 1 when programming the other ECON register data. It is always programmed seperately during the secure operation.* When programming the security bit the other ECON bits (0-4,6,7) are set to 0 for program, and only bit 5 is verified serially on DOUT (refer to timing diagram for details; The COMPARE function is not used); A 1 indicates a secured device.

A secured device reads FF from the EPROM, and all programming is inhibited. The ECON and User Bytes can be read, but cannot be programmed.

5.6 DOWNLOADING A HEX FILE TO THE PROGRAMMER

HEX files in 8 bit INTEL format should be the default format. In addition to the EPROM data, the file will include the ECON and USER BYTES data with the appropriate RAM address locations for the device EPROM size.

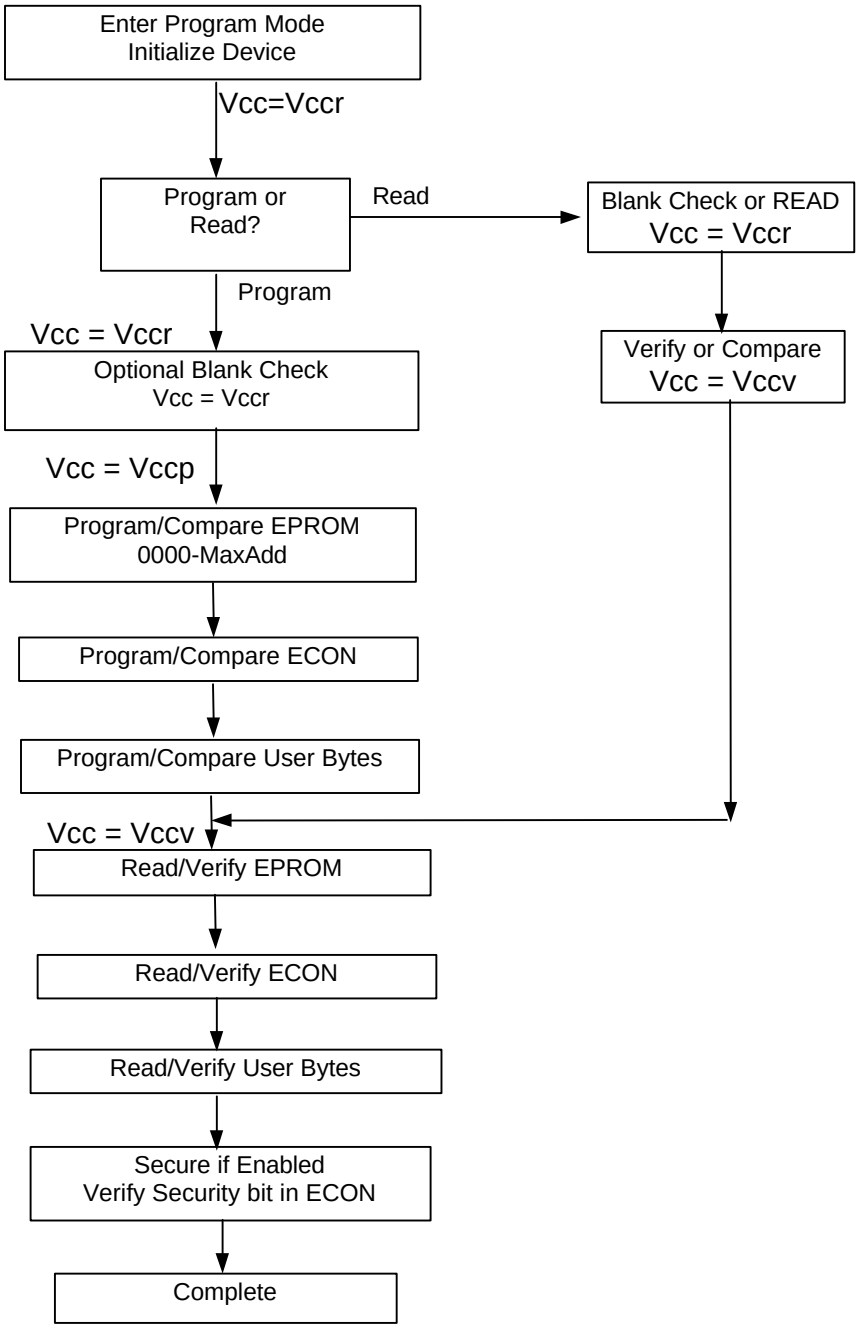
5.7 CREATING A HEX FILE

HEX files in 8 bit INTEL format should be the default format. In addition to the EPROM data, the file must include the ECON and USER BYTES data with the appropriate address locations for the device.

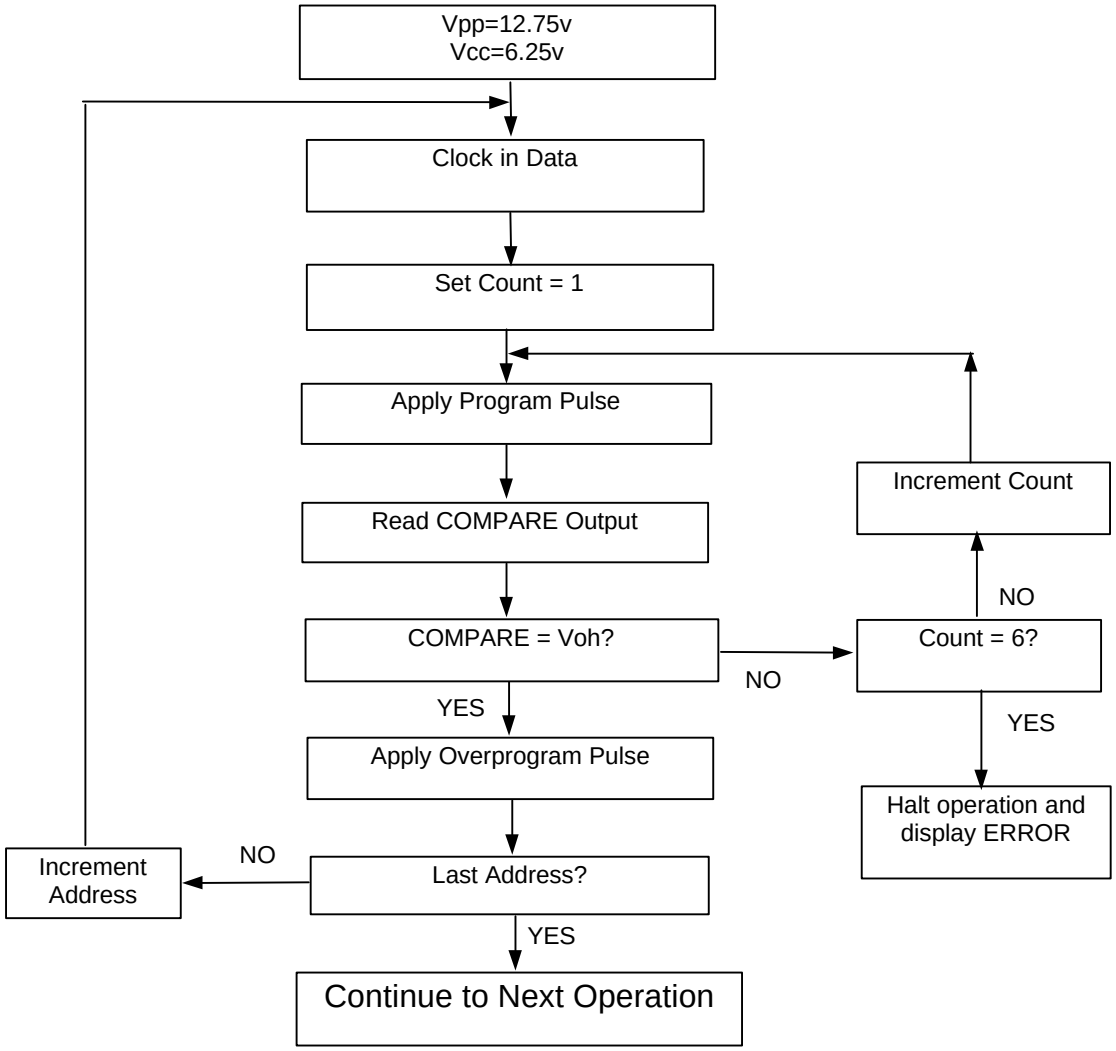
5.8 CHECKSUMS

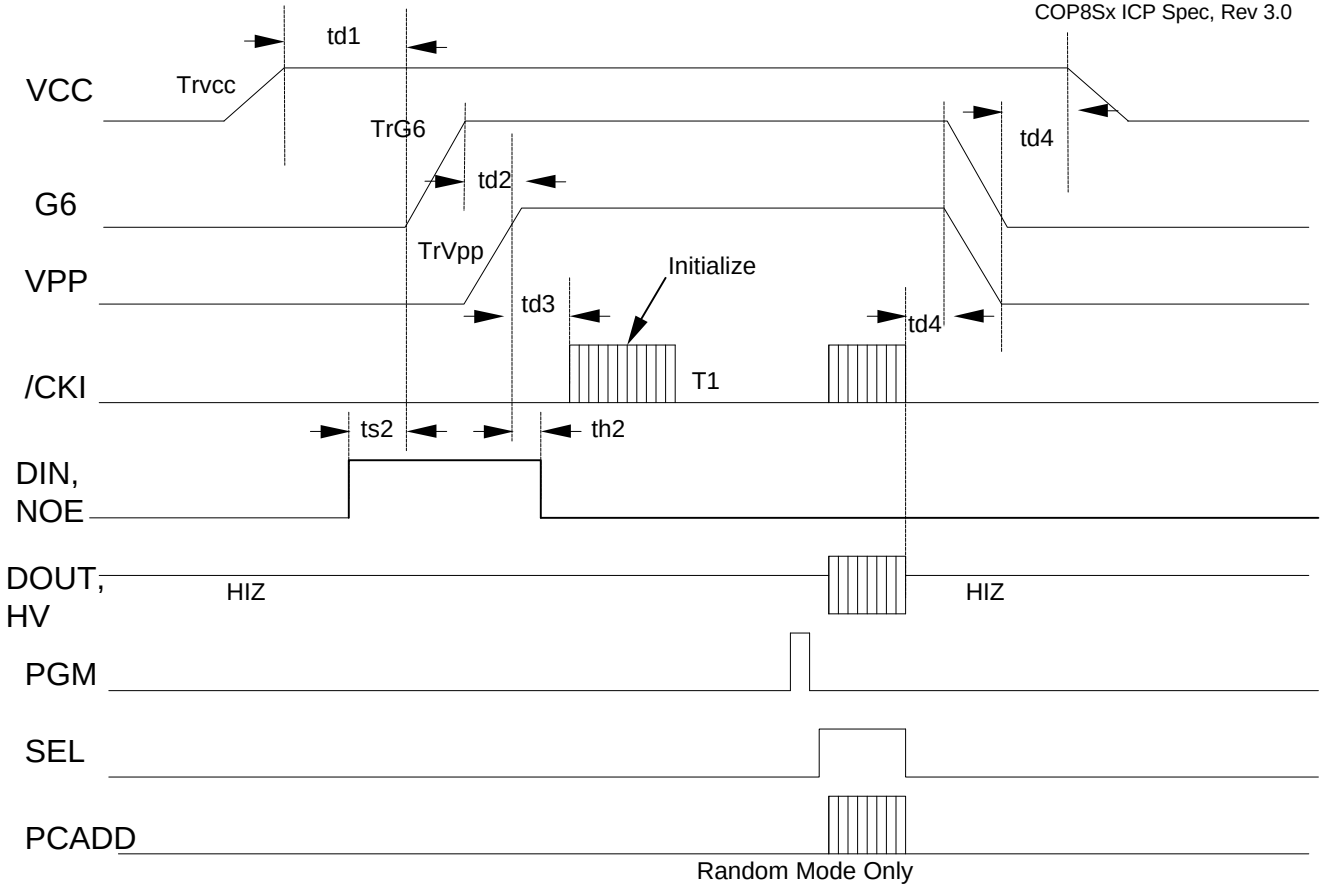
The ECON data (With bit 7 always set to 0, and including the Security bit 5), and the USER BYTES data (Default of 00) must always be included in the checksum calculation. Checksums should be displayed at the completion of any operation, including: DOWNLOAD; READ; PROGRAM; VERIFY

COP8SAx7
Program / Read Flow

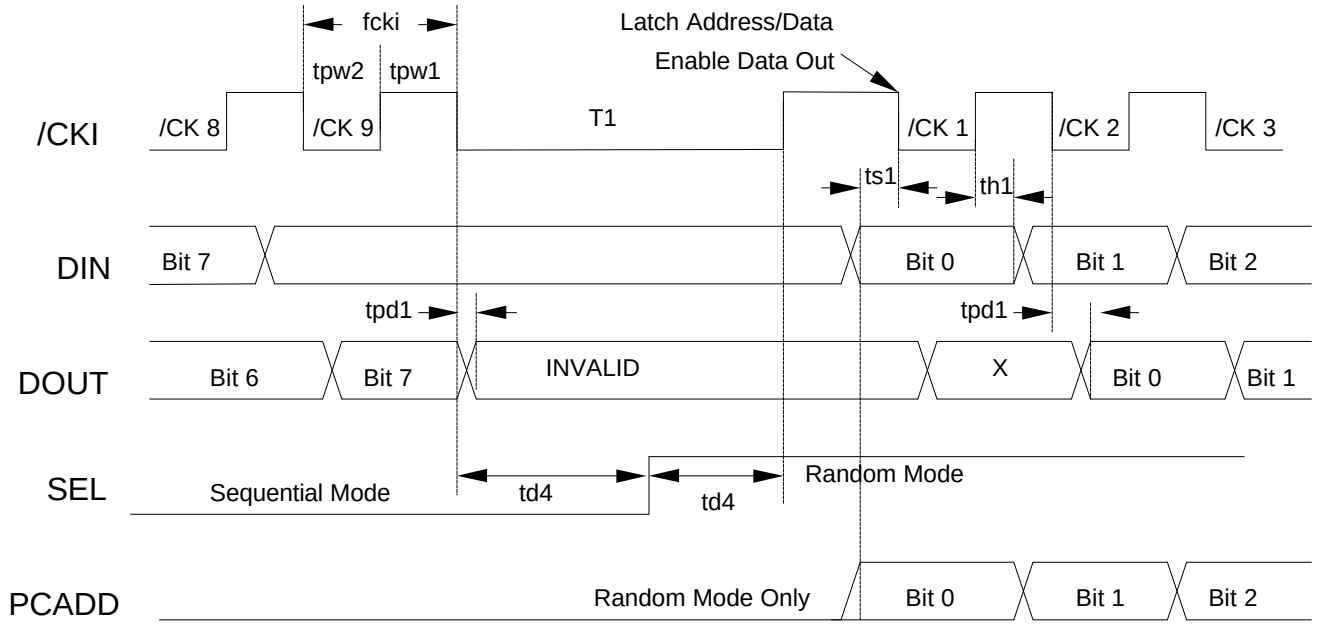


COP8SAx7
Cell Program/Compare Algorithm

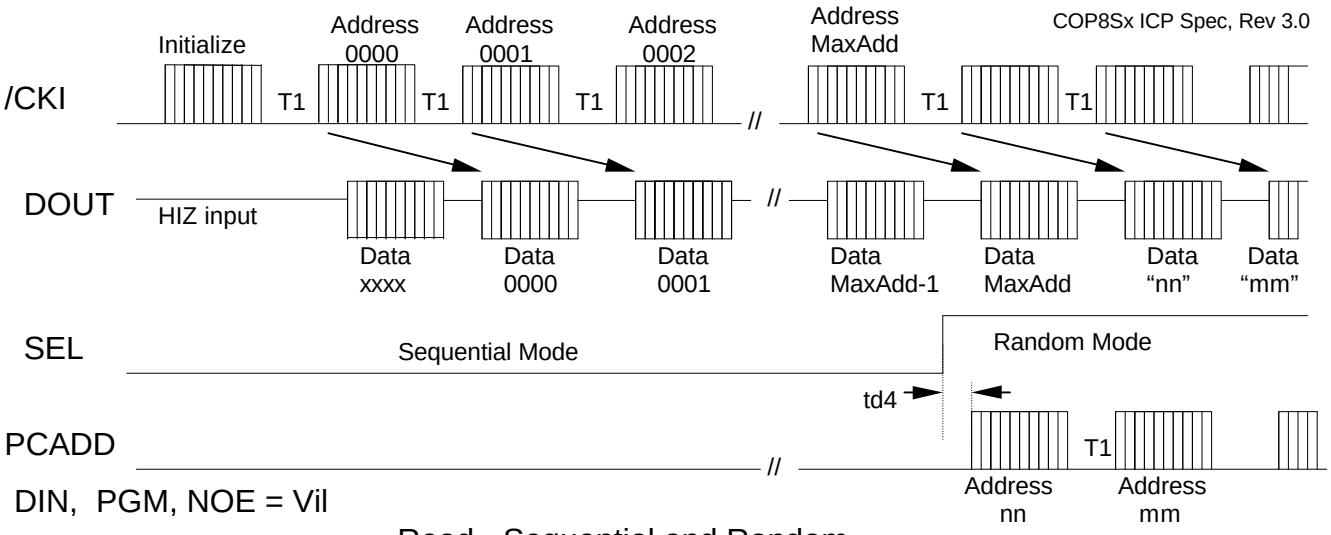




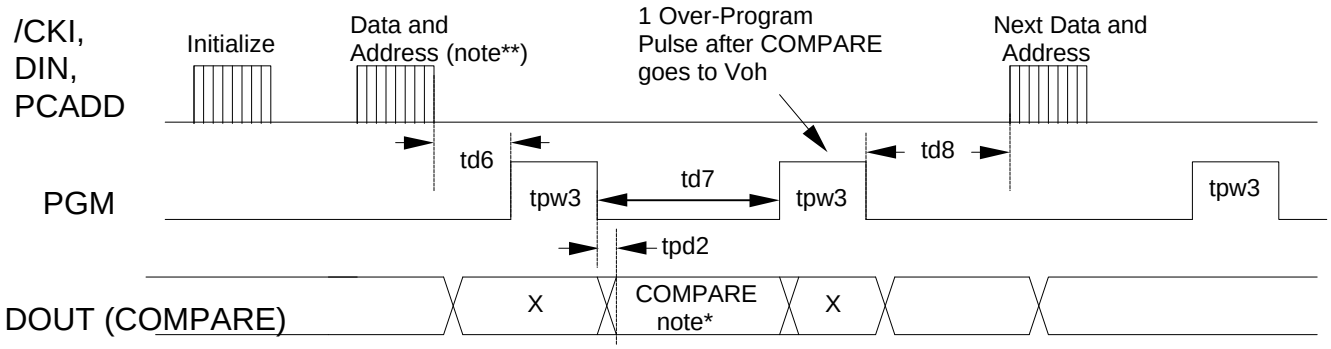
Enter/Exit Programming Mode



Data, Address, Clock Timing



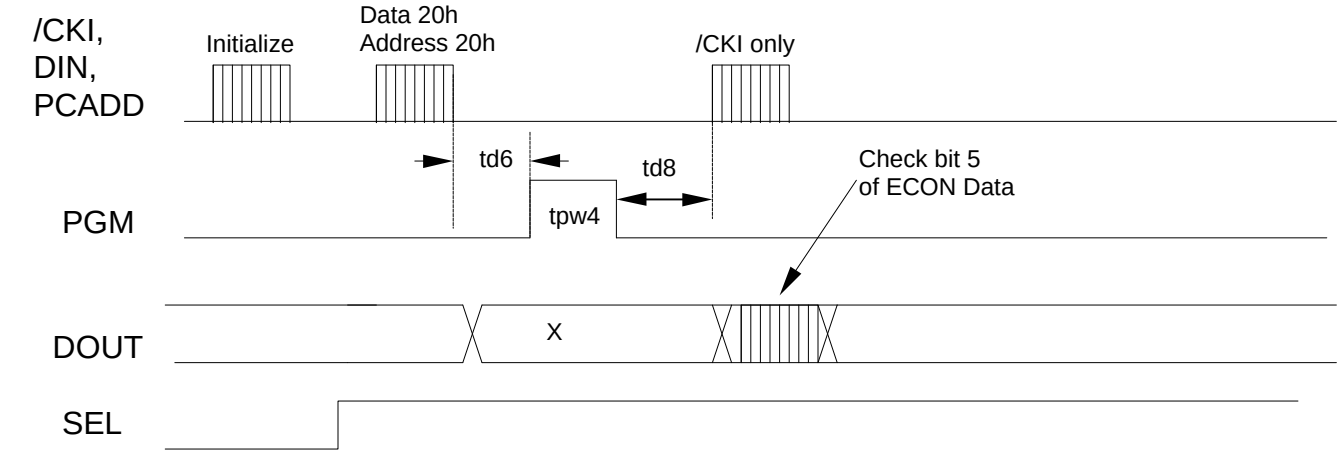
Read - Sequential and Random



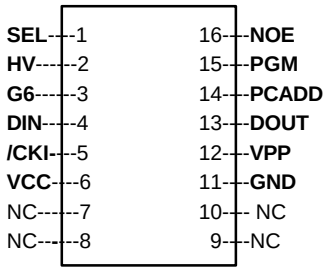
*note: If Vol, repeat tpw3 up to 6 times before rejecting

**note: First Address=0000 in Sequential Mode. PCADD input is used for Random Mode only (When SEL=Vih)

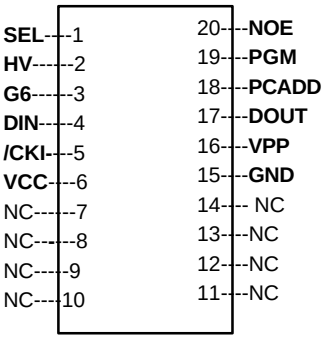
Program / Compare Timing



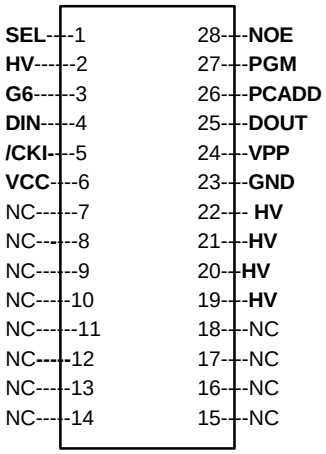
Security Program and Verify



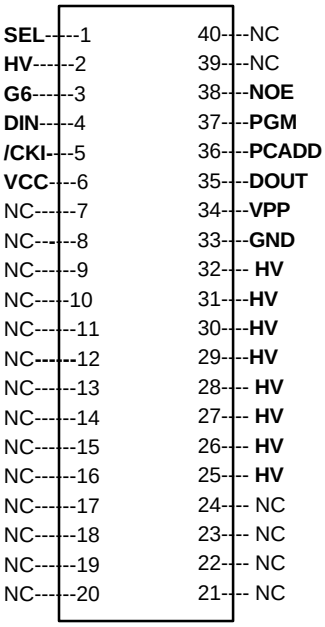
COP8S/Gxx7
16pin



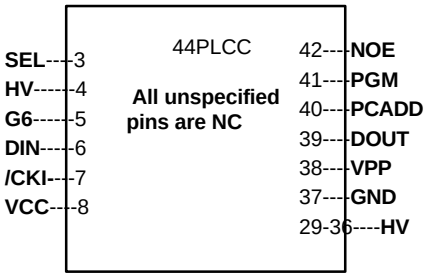
COP8S/Gxx7
20pin



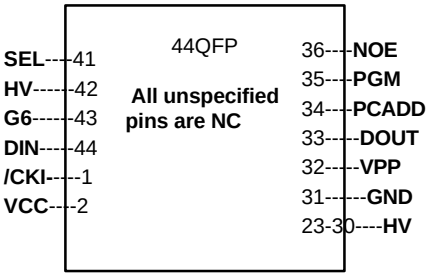
COP8S/Gxx7
28pin



COP8S/Gxx7
40pin



COP8S/Gxx7



COP8S/Gxx7

Device Pinouts

6.0 PROGRAMMING CIRCUIT IMPLEMENTATION NOTES

The COP8SAx7 devices are designed to program quickly, reliably, and without damage in less than ideal programming environments, but when constructing a programming circuit (For a personal programmer, as well as for ICP), it is very important to pay attention to some details that will insure that you obtain the best programming results possible.

6.1 PROGRAMMER VOLTAGES AND TIMING

1. VPP, G6, and VCC rise times are important. They must rise cleanly from 0v to voltage without noise or over-voltage spiking. If necessary, VPP and G6 voltages can first step to Vih then rise to high voltage, but they must rise cleanly and quickly through the range of 4v-12v.

2. VPP and G6 levels should not exceed 13v at any time, but internal protection circuits will prevent damage from overshoot or occasional spiking of less than a few nanoseconds. VCC max is 7v, and internal protection circuits will prevent damage from occasional spiking.

If your supplies have ripple, or some sagging during the programming pulse, set the levels to the high end of the spec and you should be OK.

3. When Vcc is ON, and G6/VPP are at Vil, the device is in its normal RESET state with all outputs except D0-7 at HiZ. When VPP is at Vih, the device is out of reset and may drive some pins. Care must be taken to minimize this period (Until you raise G6/VPP to high voltage), and to insure that the programming equipment is not overdriving the device outputs.

4. All timing specs are minimums, unless noted in the specification table. Rise times and a few timing periods are time sensitive, but generally there is no hurry (Except in end-of-line ICP, where time is money!). As long as inputs are clean (no noise) and stable, long delays affect only the overall cycle time.

If your timing reference is unstable, make the programming pulse longer than the spec to prevent under programming.

5. Vih levels are important, especially with /CKI. The /CKI rise time should be clean, fast, and to proper minimum input levels. Vih should not exceed Vcc at any time, but internal protection circuits will prevent damage if Vih levels exceed Vcc.

6.2 PHYSICAL DESIGN CONSIDERATIONS

Proper design of the programming assembly and isolation circuits is critical to good programming yields. Noise and ground-bounce (Disparity in grounds) are the main enemies of device programming, along with poor voltage integrity at the device pins.

1. Better designs have active drivers and voltage sources very close to the programming socket, with mechanical ground connections and large power/ground traces and connections.
Continued……

2. Better designs have independent, regulated power supply sources instead of relying on a PC connection for power.

3. For ICP, all cabling from the actual programming signal generating circuits to the device should be shielded (Twisted pairs are best), and kept as short as possible. A typical ATE cable is a good model.

4. Passive and active electronics can be mounted at the end of the cable to insure clean clock signals, good rise times, and to eliminate or reduce noise. Separate voltage sources can be connected directly to the electronics at the cable head to insure good voltage regulation over longer cable assemblies.

3. Ground and Power connections should have extra large connections, wiring, and traces.

4. Mechanically and electrically sound Power and Ground connections between the isolated device on the PCB and the programming electronics are essential. These connections (or connectors) should be as close to the actual device as possible.

6.2 PROGRAMMING CIRCUIT VERIFICATION

It is CRITICAL that all voltages and waveforms are thoroughly verified **AT THE ACTUAL DEVICE SOCKET** under actual read and programming conditions. **In an ICP environment, failure to thoroughly confirm the integrity of the programming signals at the device pins, and repeatedly test the production operation can be an expensive error!**

1. For in-circuit applications, the programming system should be connected to an actual production PCB at the manufacturing site to be used. All tests must be done at the device pins.

2. A fast storage scope with voltage level triggering is recommended for testing for noise and voltage spikes. Add capacitive filtering to your programming connections, or device socket, to correct problems or adjust voltage rise times.

3. Make sure you check ALL pins for any problems, especially the NC and HV pins.

4. Test program some test devices or PCBs REPEATEDLY (5 or 10 times) to insure that there is no cumulative damage being done to the device. Take a programmed PCB and confirm it still is 100% operational in your test environment. (You might consider doing some life test as well to detect any subtle problems caused by programming damage to the device).

5. Finally, make a pilot run of at least 100 pcs or PCBs, and confirm again that they all pass your internal testing. This will also expose any obvious programming yield issues.

6. All voltages and timing at the programming socket should be periodically re-confirmed, or before any large batch operations. You should implement some QA procedures to insure that the programming operation will be working without problems.

7.0 IN-CIRCUIT PROGRAMMING METHODOLOGY

There are several methods that can be used to physically implement end-of-line, in-circuit programming. Depending on your resources and quality expectations, one may be clearly preferable. More detailed application notes are available for some solutions.

7.1 PCB EDGE CONNECTOR WITH STAND-ALONE BOARD PROGRAMMER

This approach requires an extensive PCB design effort (New PCB form factor and layout), but may be a very cost effective manufacturing solution (Especially for longer programming times). The PCBs are hand mounted into a customized, multi-socket programming system (Similar to a gang EPROM programmer), and programmed in parallel. Throughput is high and total test time is low

This approach offers a high quality programming solution, with little physical impact on your existing production line. Implementation requires minimum expertise from the end-user. Redundancy is easy to implement with a backup programmer. There are a number of experienced programmer manufacturers that are available to consult on PCB design, and offer end-of-line programming systems

7.2 PCB SURFACE-MOUNT CONNECTOR WITH STAND-ALONE ICP PROGRAMMER

This approach requires less PCB design effort (You only add a surface mounted connector and some isolation circuitry), but may ultimately be a more costly manufacturing solution due to a lower programming throughput (Especially if programming times are long). A stand-alone programmer with an ICP connector is manually plugged into the PCB connector. Devices are programmed one at a time.

This approach offers a good quality programming solution, with little physical impact on your existing production line. Implementation requires some expertise from the end-user to create/debug/maintain the programming assembly. Redundancy is easy to implement with a backup programmer. There are a number of experienced programmer manufacturers that are available to consult on PCB design, and offer end-of-line programming systems for this application.

7.3 PCB SURFACE-MOUNT CONNECTOR WITH ATE

This approach requires less PCB design effort (You only add a surface mounted connector and some isolation circuitry), but could be a much more costly manufacturing solution unless you already incorporate ATE in your product's final test. A customized ATE system with an ICP connector is manually (or by machine) plugged into the PCB connector. Devices are programmed one at a time.

This approach offers a good quality programming solution, but will impact your existing production line. Implementation requires significant expertise from the end-user to modify the ATE system and create/debug/maintain the programming functions and assemblies. Redundancy is probably not feasible due to cost. There are few experienced ATE manufacturers that are available to consult on designing this type of end-of-line programming system.

7.4 BED-OF-NAILS ATE

This approach requires the least amount of PCB design effort (You only add some isolation circuitry), but could be a much more costly manufacturing solution unless you already incorporate bed-of-nails ATE in your product's final test. A customized ATE system connects to the PCB in normal bed-of-nails fashion. Devices are programmed one at a time.

This approach offers a high quality programming solution, with virtually no impact on your existing production line. Implementation requires significant expertise from the end-user to modify the ATE system and create/debug/maintain the programming functions. Redundancy is probably not feasible due to cost. There are few experienced ATE manufacturers that are available to consult on designing this type of end-of-line programming system.

7.5 DEVICE LEAD CLIP-ON CONNECTION WITH ATE OR PROGRAMMER

This approach requires the least amount of PCB design effort (You only add some isolation circuitry), and can be very-low cost to implement. But it could be a much more costly manufacturing solution because of expected higher ppm fallout. A custom programming cable (from ATE or some other programmer) is manually clipped on to the device leads. Devices are programmed one at a time.

This approach offers a poor quality programming solution (Primarily due to inconsistent, unreliable device contacts), but with minimum impact on your existing production line. Implementation requires some expertise from the end-user to create/debug/maintain the programming assembly. Redundancy is easy with a backup programmer. This method is strongly discouraged by industry experts, and is NOT supported by National.

8.0 ICP PROGRAMMING VENDORS AND CONSULTANTS

The following vendors are experienced in providing ICP and end-of-line programming consulting, and programming solutions:

Data I/O Corp; USA and world-wide. www.data-io.com - Board-level programming systems and other production line solutions.

Stag Programmers; UK and world-wide. www.stag.co.uk - Board -level programming systems and other production line solutions.

Teradyne; USA and world-wide. - Bed-of-nails ATE system with programming capability.