

# **P89LPC900**

## **In-Circuit Programming (ICP) Specifications**

Programming Specification

2005 Sep 26

**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

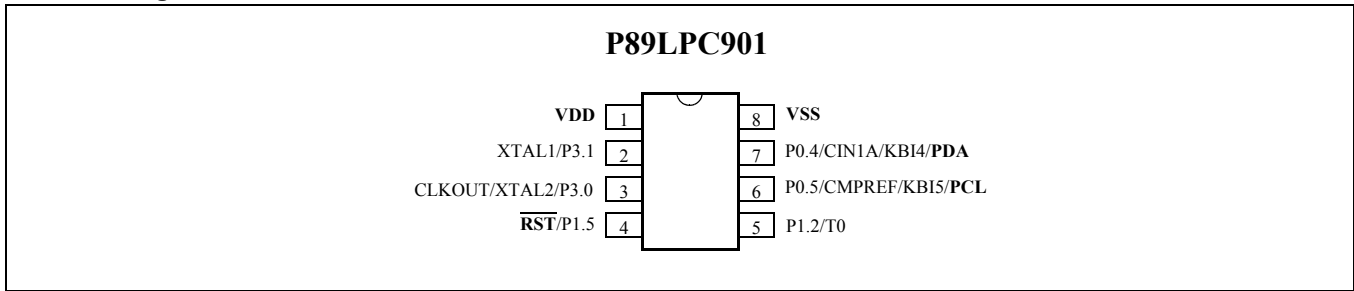
# Table of Contents

<b>Pin Configurations .....</b>	<b>3</b>
Pin Descriptions .....	13
Device ID bytes .....	14
<b>Programming mode .....</b>	<b>15</b>
Getting into the serial programming mode .....	15
Programming commands .....	16
<b>Checking status information .....</b>	<b>17</b>
<b>Flash operating sequences .....</b>	<b>18</b>
Loading the full Page Register .....	18
Loading the Page Register partially .....	19
Programming User Code Memory.....	20
Erasing all sectors (global erase).....	21
Erasing a single sector .....	22
Erasing a single page.....	23
Calculate Global CRC .....	24
Calculate Sector CRC .....	25
Reading Configuration, Boot Vector, Status Byte, Security Bits, Signature Bytes .....	26
Writing Configuration, Boot Vector, Status Byte, and Security Bits.....	28
Writing CCP Clear Configuration Protection .....	29
<b>Calculating CRC .....</b>	<b>30</b>
<b>AC timings .....</b>	<b>31</b>
Getting into the serial programming mode .....	31
Timing data shifting .....	32
<b>Revision History .....</b>	<b>33</b>

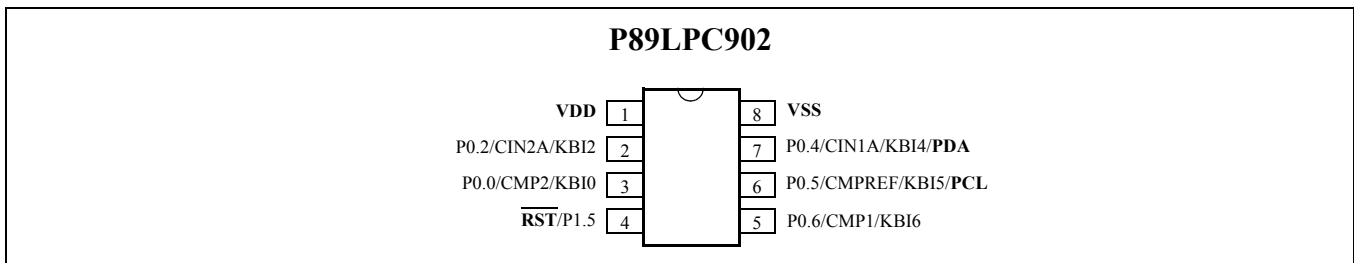
**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

# 1.0 Pin Configurations

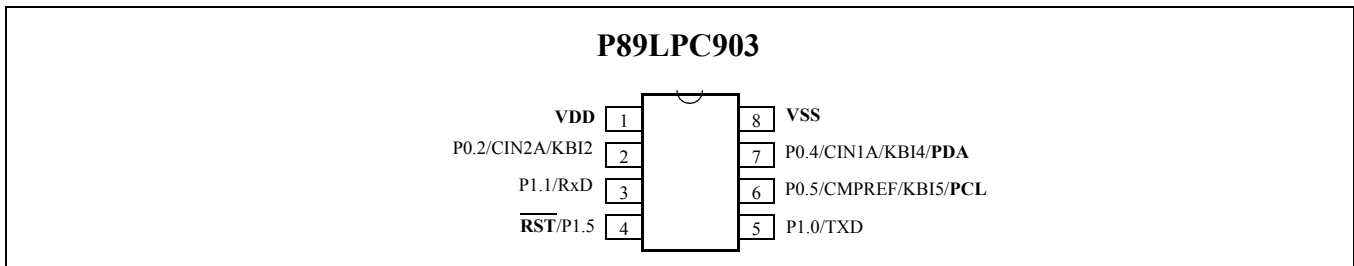
## 8-Pin Packages



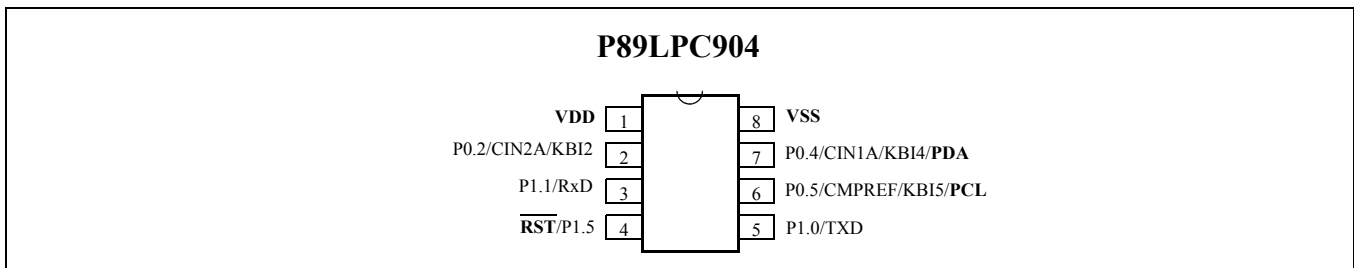
**Figure 1: P89LPC901 pinout**



**Figure 2: P89LPC902 pinout**



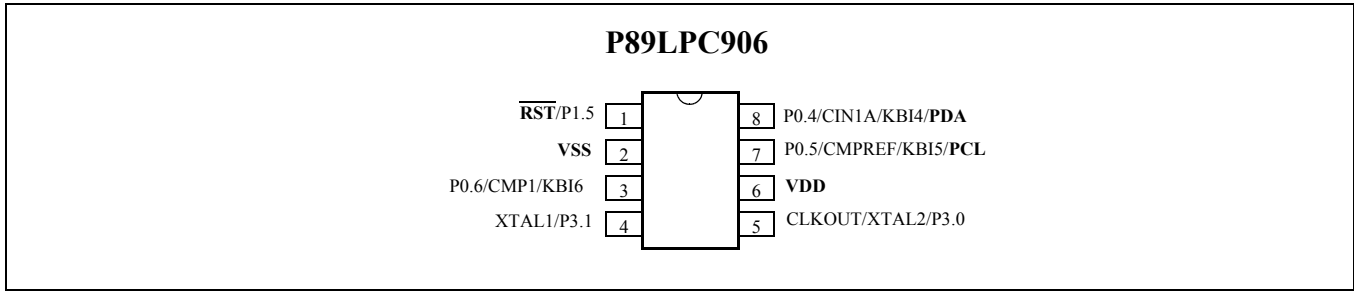
**Figure 3: P89LPC903 pinout**



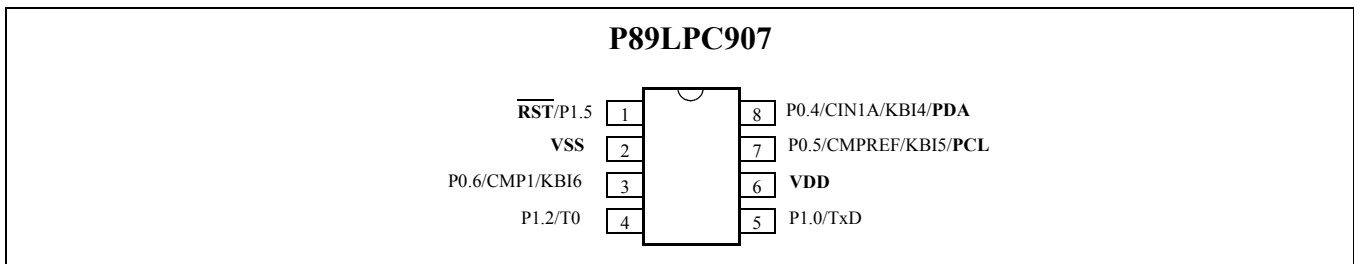
**Figure 4: P89LPC904 pinout**

# P89LPC9xx

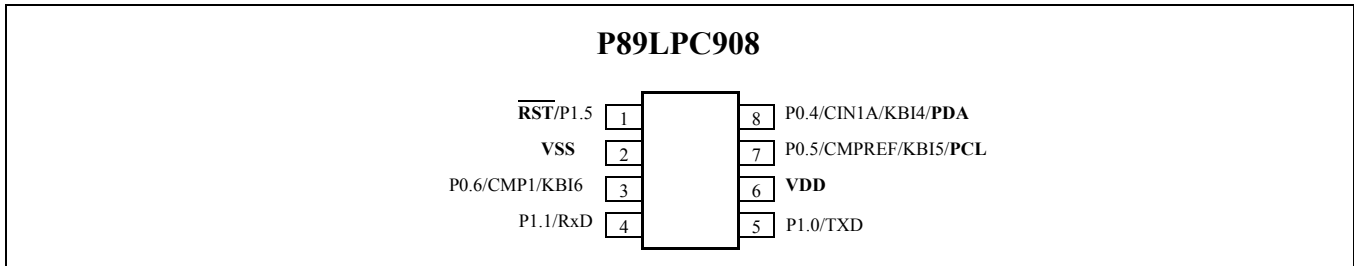
## In Circuit Programming (ICP) Specifications



**Figure 5: P89LPC906 pinout**



**Figure 6: P89LPC907 pinout**

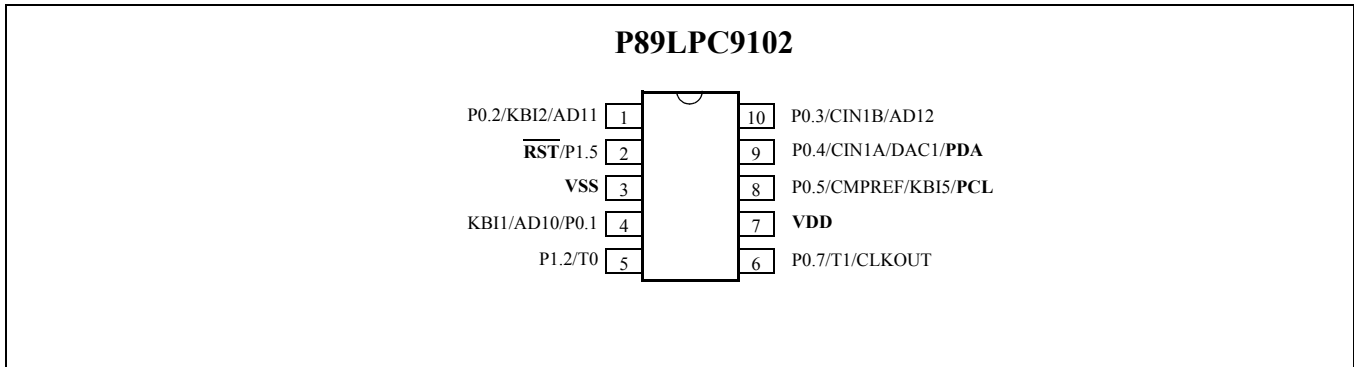


**Figure 7: P89LPC908 pinout**

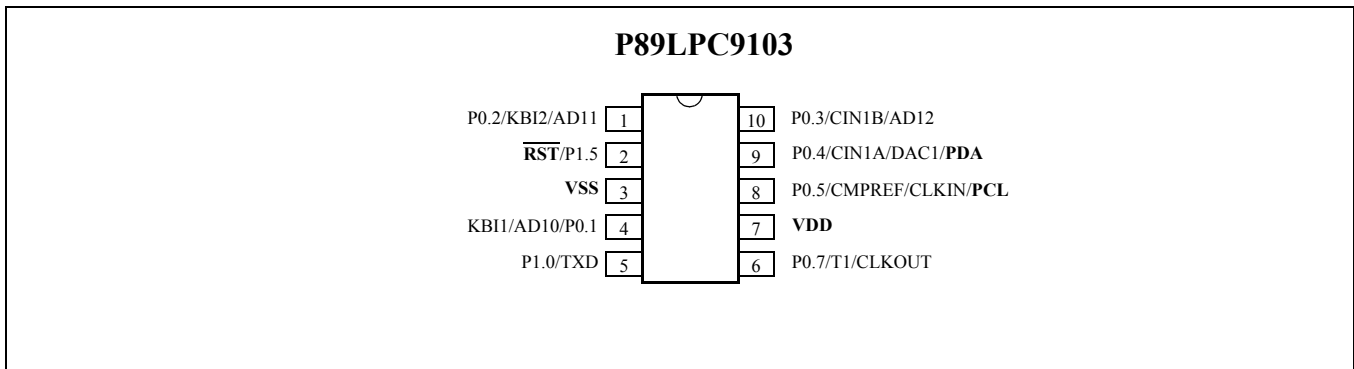
# P89LPC9xx

## In Circuit Programming (ICP) Specifications

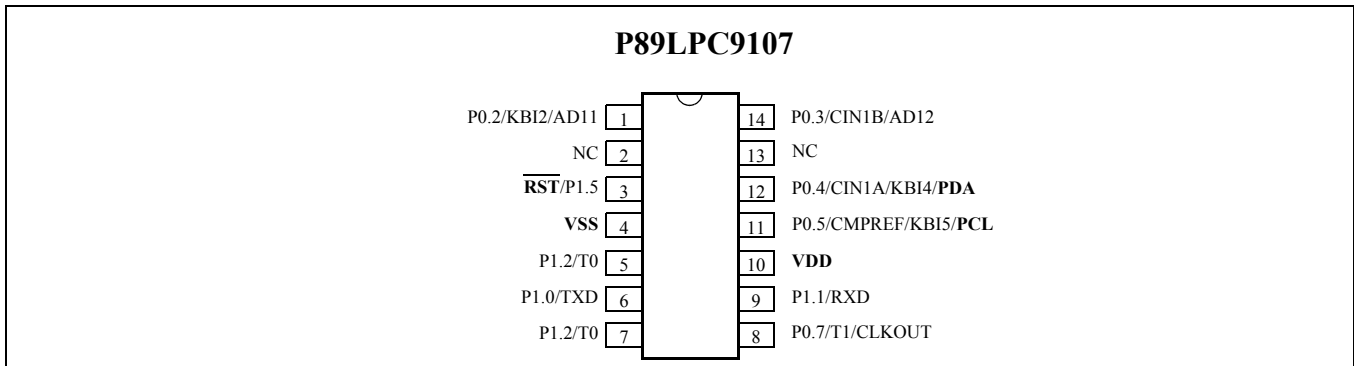
### 10-Pin Packages



**Figure 8: P89LPC9102 pinout**



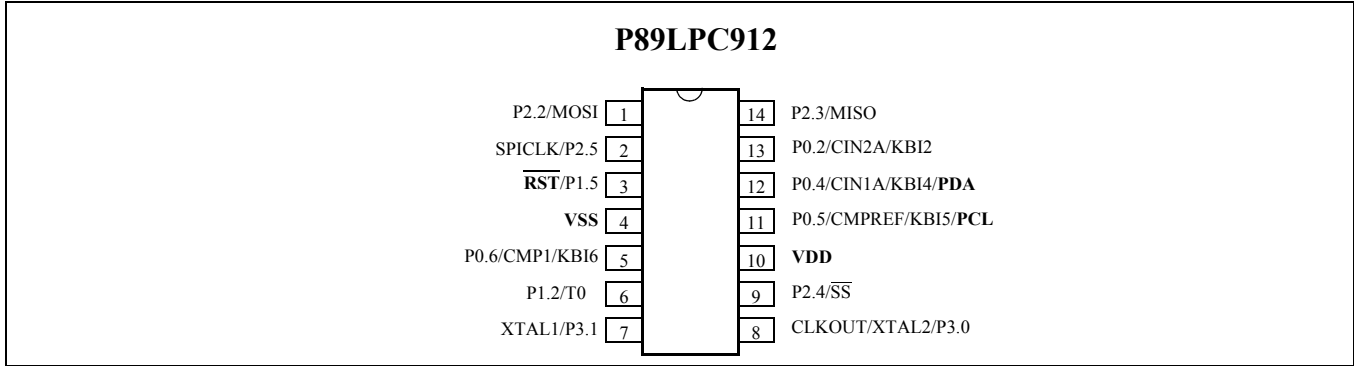
**Figure 9: P89LPC9103 pinout**



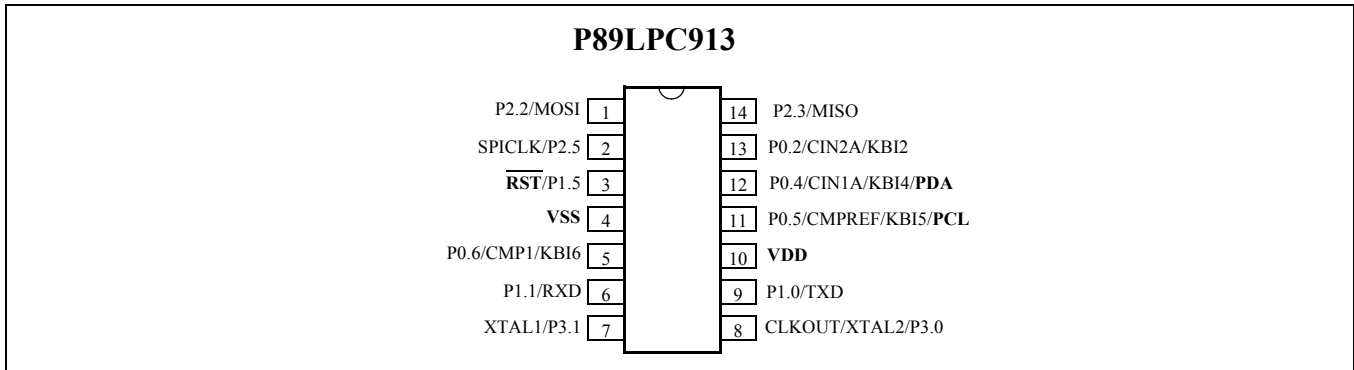
**Figure 10: P89LPC9107 pinout**

# P89LPC9xx

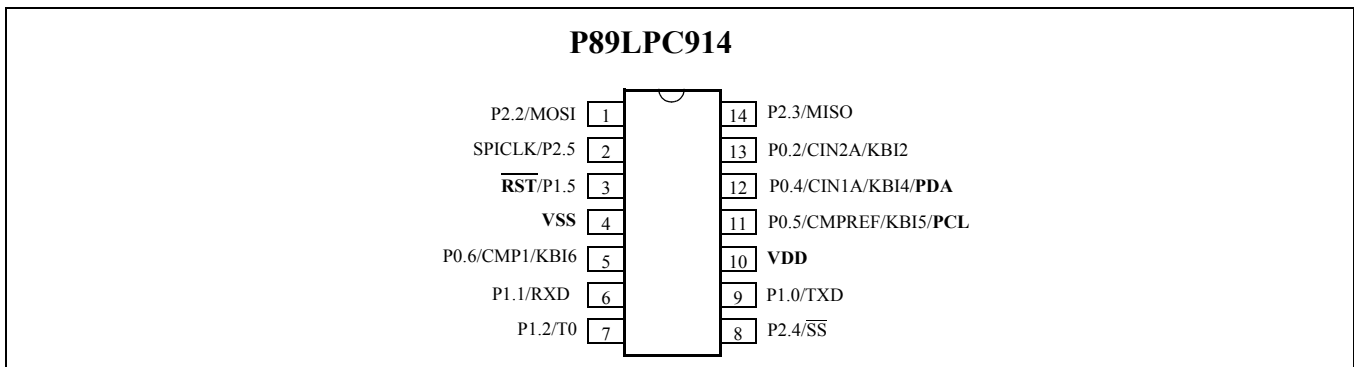
## In Circuit Programming (ICP) Specifications



**Figure 11: P89LPC912 pinout**



**Figure 12: P89LPC913 pinout**



**Figure 13: P89LPC914 pinout**

# P89LPC9xx

## In Circuit Programming (ICP) Specifications

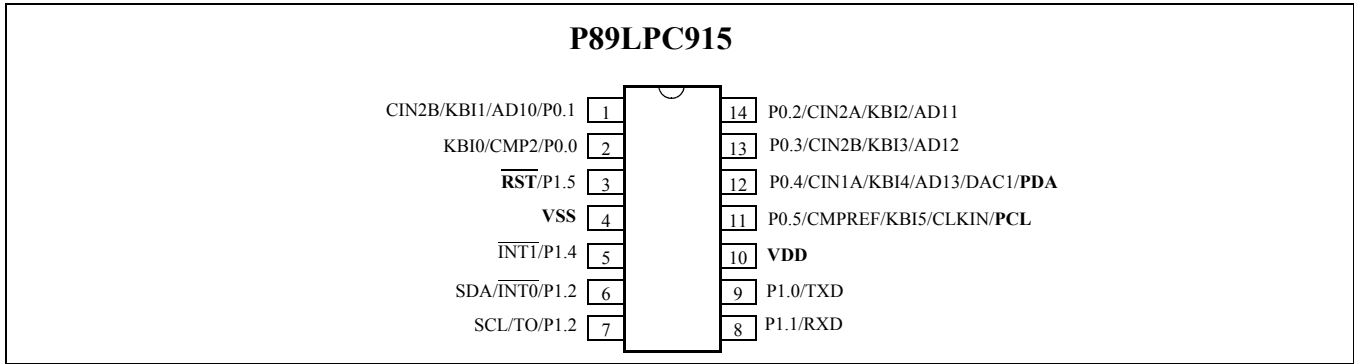


Figure 14: P89LPC915 pinout

### 16-Pin Packages

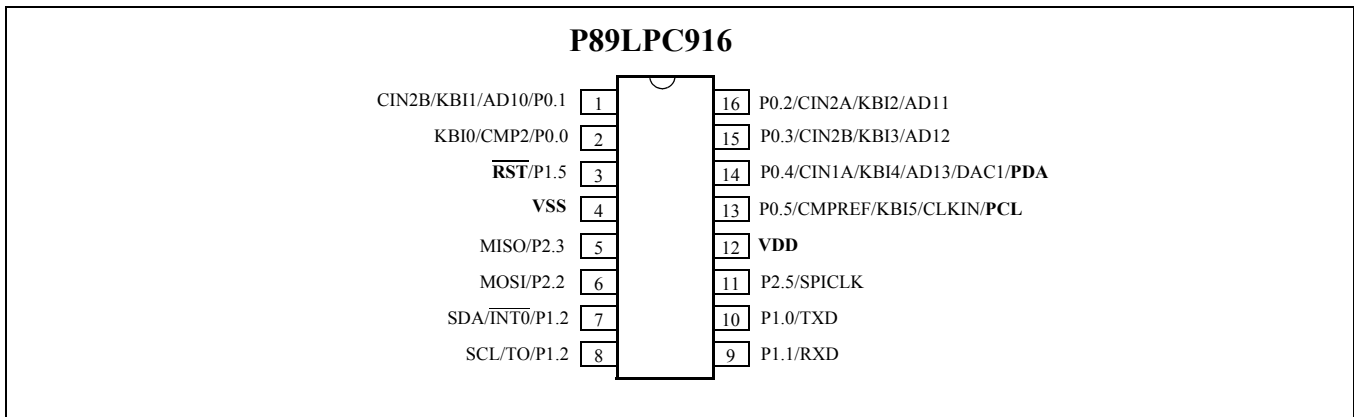


Figure 15: P89LPC916 pinout

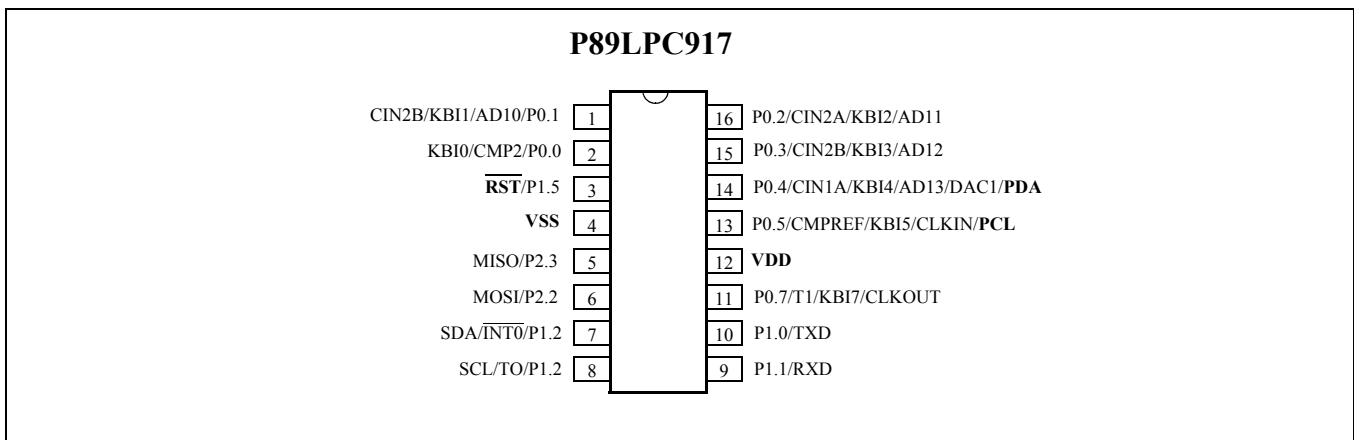


Figure 16: P89LPC917 pinout

# P89LPC9xx

## In Circuit Programming (ICP) Specifications

### 20-Pin Packages

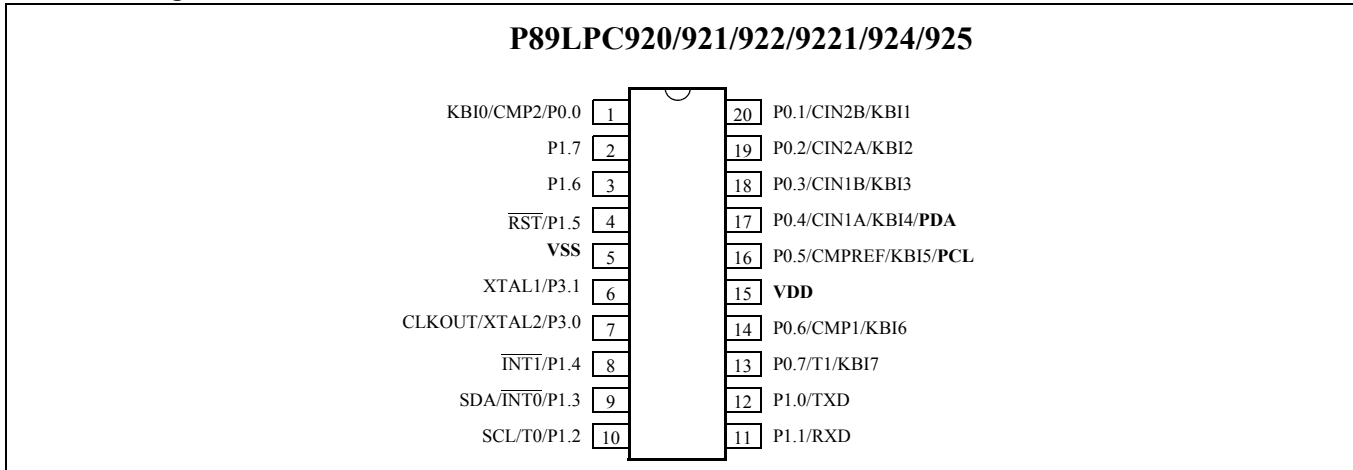


Figure 17: P89LPC922/921/920/924/925 pinout

### 28-Pin Packages

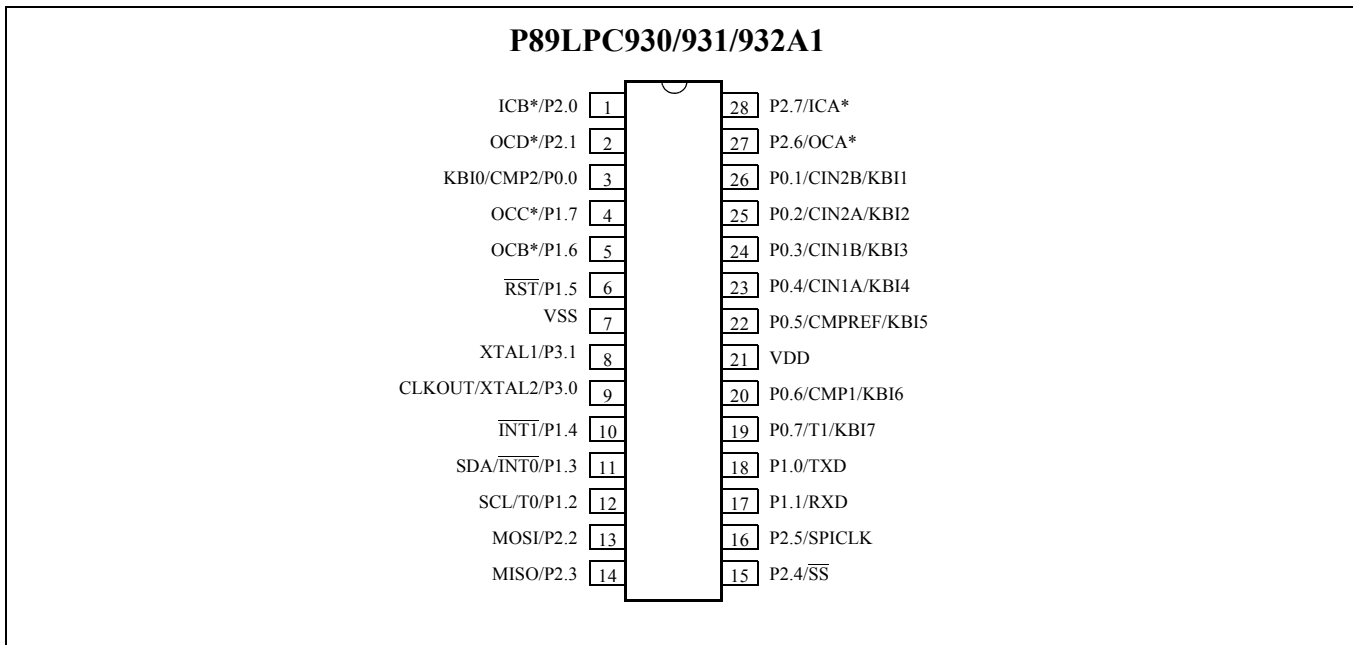
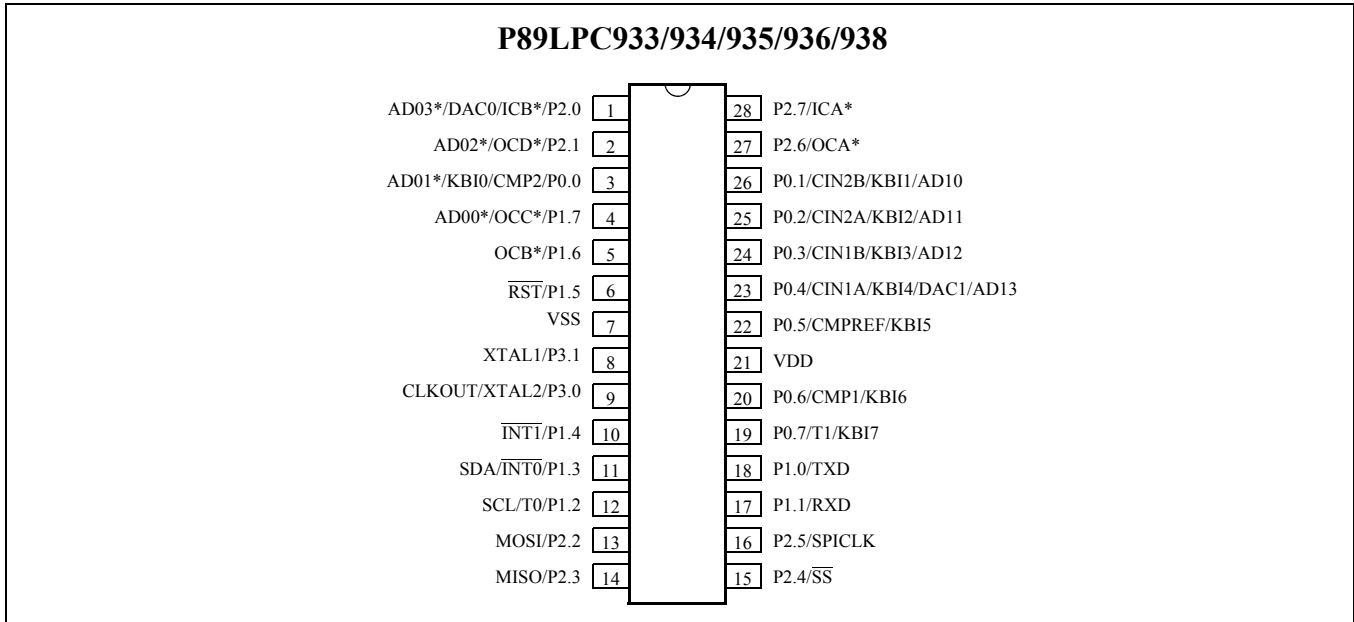


Figure 18: P89LPC930/931/932A1

\*Only on the LPC932A1

# P89LPC9xx

## In Circuit Programming (ICP) Specifications

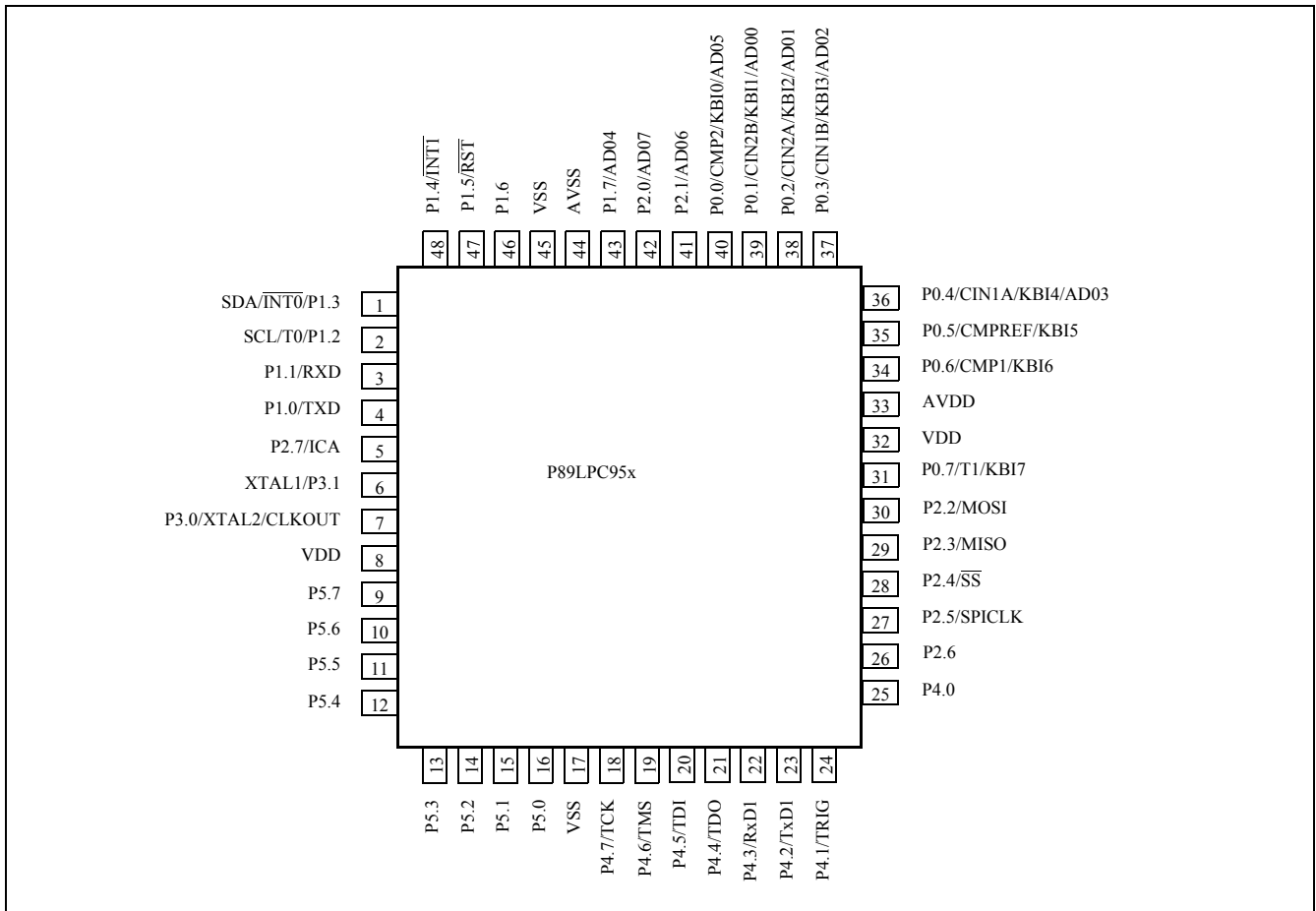


**Figure 19: P89LPC933/934/935/936/938 pinout**

\*Only on the LPC935

# P89LPC9xx

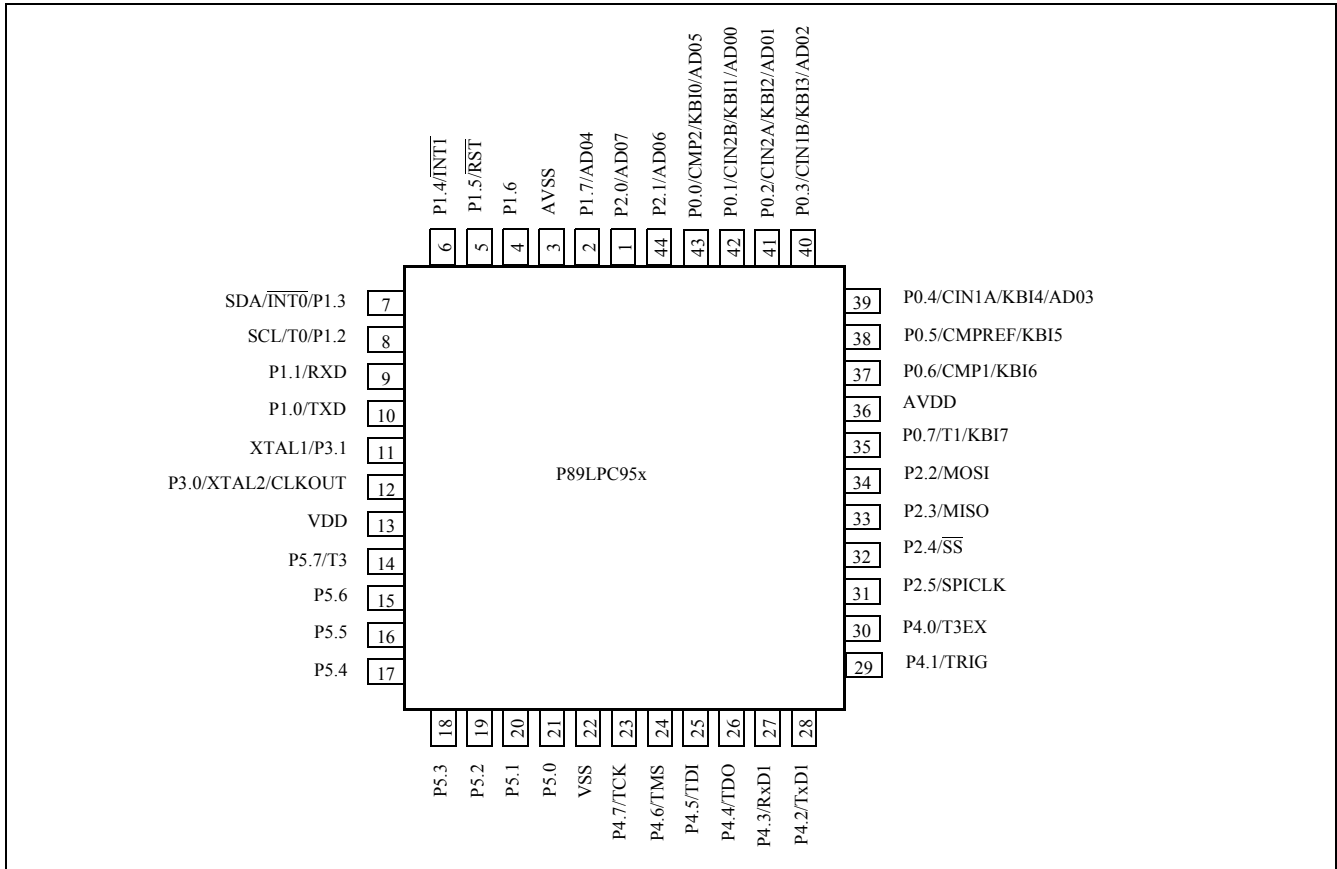
## In Circuit Programming (ICP) Specifications



**Figure 20: P89LPC952 LQFP48 pinout**

# P89LPC9xx

## In Circuit Programming (ICP) Specifications



**Figure 21: P89LPC952 PLCC44 pinout**

# P89LPC9xx

## In Circuit Programming (ICP) Specifications

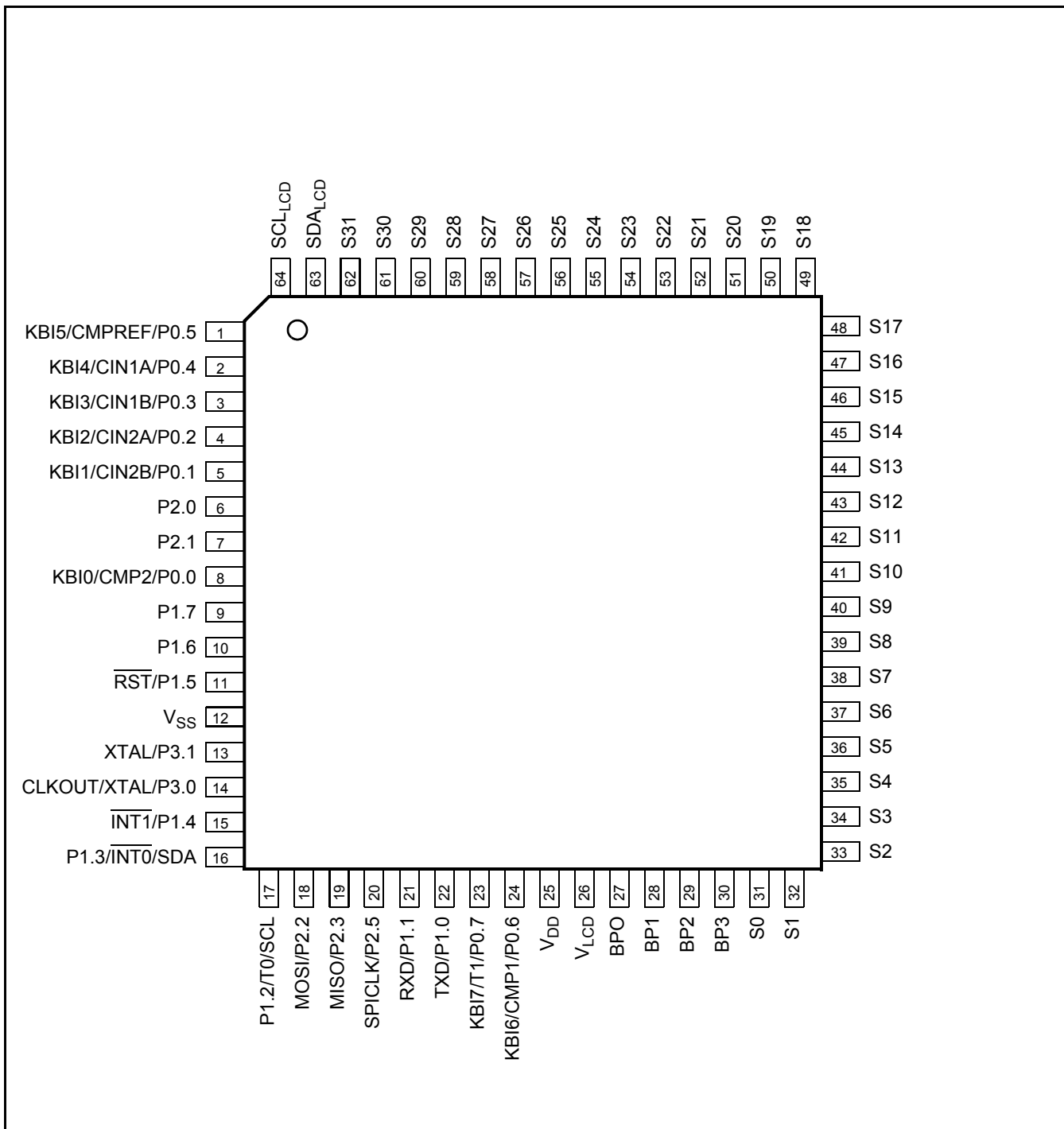


Figure 22: P89LPC9401

**P89LPC9xx****In Circuit Programming (ICP) Specifications****1.1 Pin Descriptions**

<b>MNEMONIC</b>	<b>TYPE</b>	<b>NAME AND FUNCTION</b>
V <sub>SS</sub>	P	Ground: 0V reference.
V <sub>DD</sub>	P	Power Supply: 3V
PCL	I	Serial clock input for programming communication.
PDA	I/O	Serial data I/O for programming communication.
RESET	I	ICP programming entry pin

**Table 1: Programming pins used for ICP**

## P89LPC9xx

### In Circuit Programming (ICP) Specifications

#### 1.2 Device ID bytes

Device	MFGID	ID1	ID2
P89LPC901	15h	DD	0Dh
P89LPC902	15h	DD	0Fh
P89LPC903	15h	DD	10h
P89LPC904	15h	DD	21h
P89LPC906	15h	DD	11h
P89LPC907	15h	DD	12h
P89LPC908	15h	DD	13h
P89LPC9102	15h	DD	22h
P89LPC9103	15h	DD	23h
P89LPC9107	15h	DD	27h
P89LPC912	15h	DD	14h
P89LPC913	15h	DD	15h
P89LPC914	15h	DD	16h
P89LPC915	15h	DD	17h
P89LPC916	15h	DD	18h
P89LPC917	15h	DD	20h
P89LPC920	15h	DD	1Ah
P89LPC921	15h	DD	0Bh
P89LPC922/ 9221	15h	DD	0Ch
P89LPC924	15h	DD	1Bh
P89LPC925	15h	DD	1Ch
P89LPC930	15h	DD	19h
P89LPC931	15h	DD	09h
P89LPC932A1	15h	DD	1Fh
P89LPC933	15h	DD	A0h
P89LPC934	15h	DD	1Dh
P89LPC935	15h	DD	1Eh
P89LPC936	15h	DD	24h
P89LPC938	15h	DD	25h
P89LPC9401	15h	DD	26h
P89LPC952	15h	DD	28h

Table 2: ID bytes for different devices

**P89LPC9xx****In Circuit Programming (ICP) Specifications**

## 2.0 Programming mode

The programming commands are sent by the programmer through the PCL and PDA lines.

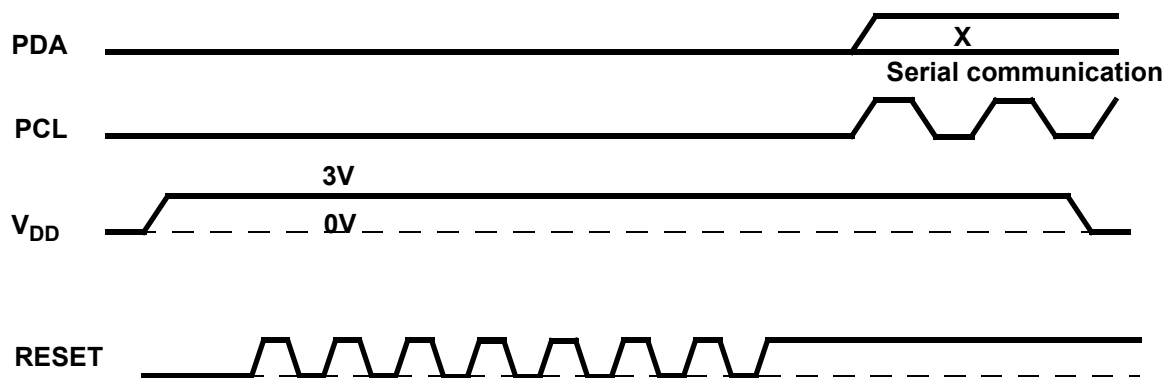
Each programming command is one byte shifted into the part by 8 clocks. The serial interface is identical to the 51's 8-bit serial UART mode 0; LSB is the first bit in the serial byte. The PCL pin is the clock input from the programmer.

The PDA pin is the data I/O. The Data is enabled on the falling edge of PCL, and is clocked on the rising edge of PCL. Data output from the part is disabled after the rising edge of PCL for the last bit in a data byte. Writing programming commands by the programmer Commands.

When the P89LPC90x is in programming mode all pins that are not used for programming are tri-stated. During programming mode the reset pin has a weak pull-up resistor.

### 2.1 Getting into the serial programming mode

To get in programming mode the RESET pin has to have a sequence of 7 pulses after the rising edge of VDD. The timing requirements have to be met to get into programming mode, otherwise the P89LPC9xx will run in normal user mode. See AC specifications (section 5) for timings.



**Figure 23: Getting into the programming mode**

## P89LPC9xx In Circuit Programming (ICP) Specifications

### 2.2 Programming commands

Programming operations can be done by shifting in commands and data in sequence. The following table shows the commands that can be used.

COMMANDS	OPCODE	FUNCTION
NOP	00H	NOP
WR_FMADRL	08H	Write address low command
RD_FMADRL	09H	Read address low command
WR_FMADRH	0AH	Write address high command
RD_FMADRH	0BH	Read address high command
WR_FMCON	0EH	Write a command to FMCON
RD_FMCON	0FH	Read a command from FMCON
WR_FMDATA_PG	14H	Write a command to FMDATA and increment address
RD_FMDATA_PG	15H	Read a command from FMDATA and increment address
WR_FMDATA	0CH	Write a command to FMDATA
RD_FMDATA	0DH	Read a command from FMDATA
WR_FMDATA_I	04H	Write a command to FMDATA and increment address
RD_FMDATA_I	05H	Read a command from FMDATA and increment address

**Table 3: Programming commands**

When the WR\_FMCON command has been sent the FM\_CON (Flash control register) can be loaded with the following commands:

FMCON Write COMMANDS	Command CODE	FUNCTION
LOAD	00H	Clear and then load the page register
PROG	48H	Program page with page register command
ERS_G	72H	Erase global command
ERS_S	71H	Erase sector command
ERS_P	70H	Erase page command
CONF	6CH	Accesses user configuration information addressed by FMADRL
CRC_G	1AH	Calculate CRC on the entire user code command
CRC_S	19H	Calculate CRC on a sector command
CCP	67H	Clear configuration protection

**Table 4: FMCON Write commands**

**P89LPC9xx****In Circuit Programming (ICP) Specifications**

---

### 3.0 Checking status information

FMCON contains status information when the RD\_FMCON command is used to read FMCON, and is updated by the last operation performed. FMCON contains the following bits:

0	OI	Operation aborted by an interrupt.
1	SV	Security Violation. Set if operation fails due to security violation.
2	HVE	High Voltage Error. Set if error detected in high voltage generation circuit.
3	HVA	High Voltage Abort. Set if high volt. aborted due to bandgap/ brownout problems.
4	---	unused
5	---	unused
6	---	unused
7	BUSY	Set while a program, erase, or other programming operation is in progress.

The Unused bits will read out as '1's. When no errors occur FMCON will read out F0h when still busy performing an operation and 70h when an operation has successfully completed.

**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

## 4.0 Flash operating sequences

### 4.1 Loading the full Page Register

The page register is loaded by performing the following sequence:

1. Activate the ICP Programming Mode, as previously described, if not already performed.
2. Shift in the WR\_FMCON opcode.
3. Shift in 00h, the LOAD command to FMCON.
4. Shift in 08h, WR\_FMADRL opcode.
5. Shift in X0 to FMADRL. This sets the destination of the first data byte to be loaded as the beginning of the page.
6. Shift in the 14h, WRFMDATA\_PG opcode.
7. Shift in X bytes of data to be loaded into the page register. FMADRL is incremented after each byte. After the last byte FMADRL rolls over to point to the beginning of the page. \*

**Note: LPC92x / LPC93x have 64 byte page registers, LPC91x / LPC90x have 16 byte page registers.**

Example of a sequence loading the Page register when in programming mode:

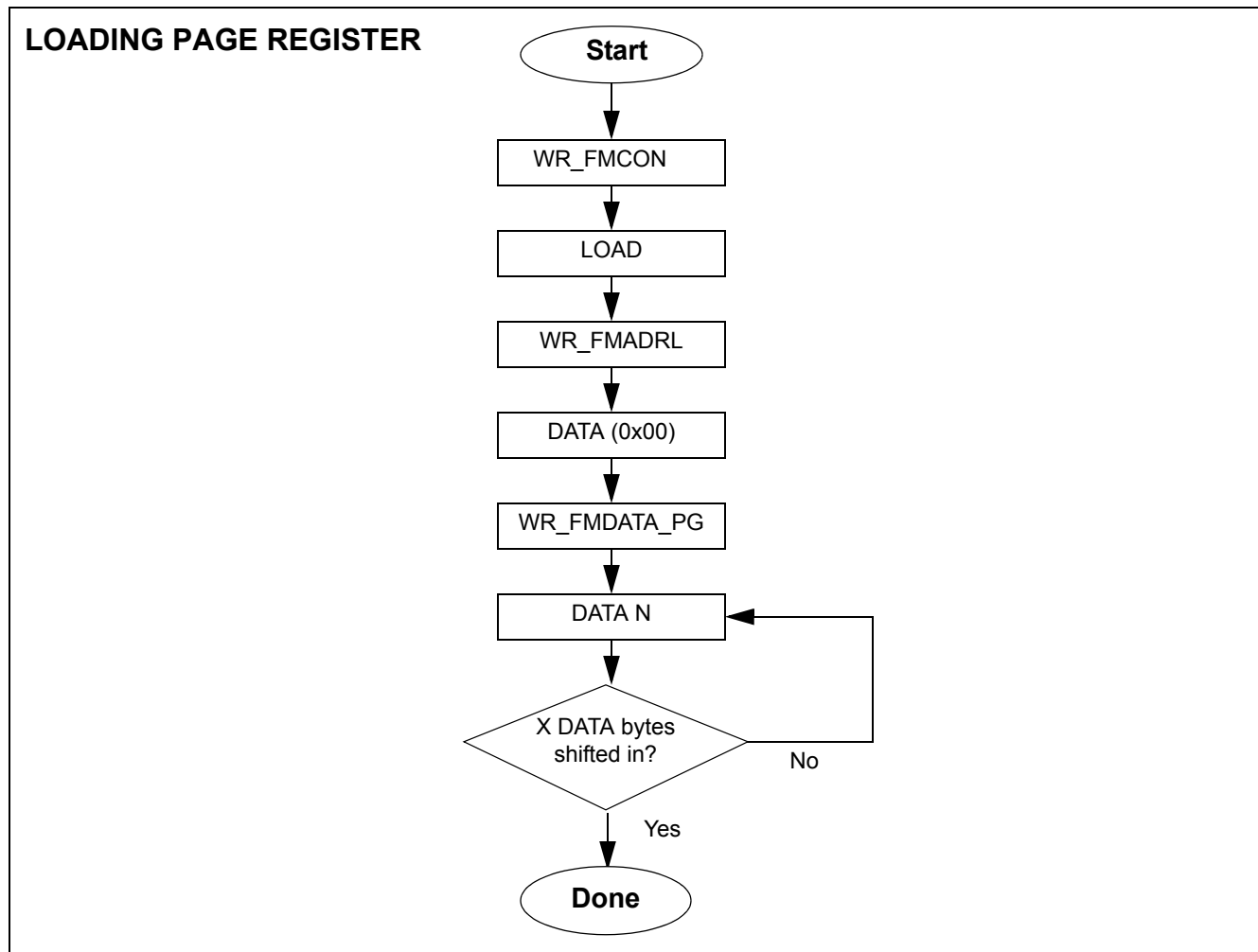


Figure 24: loading page register

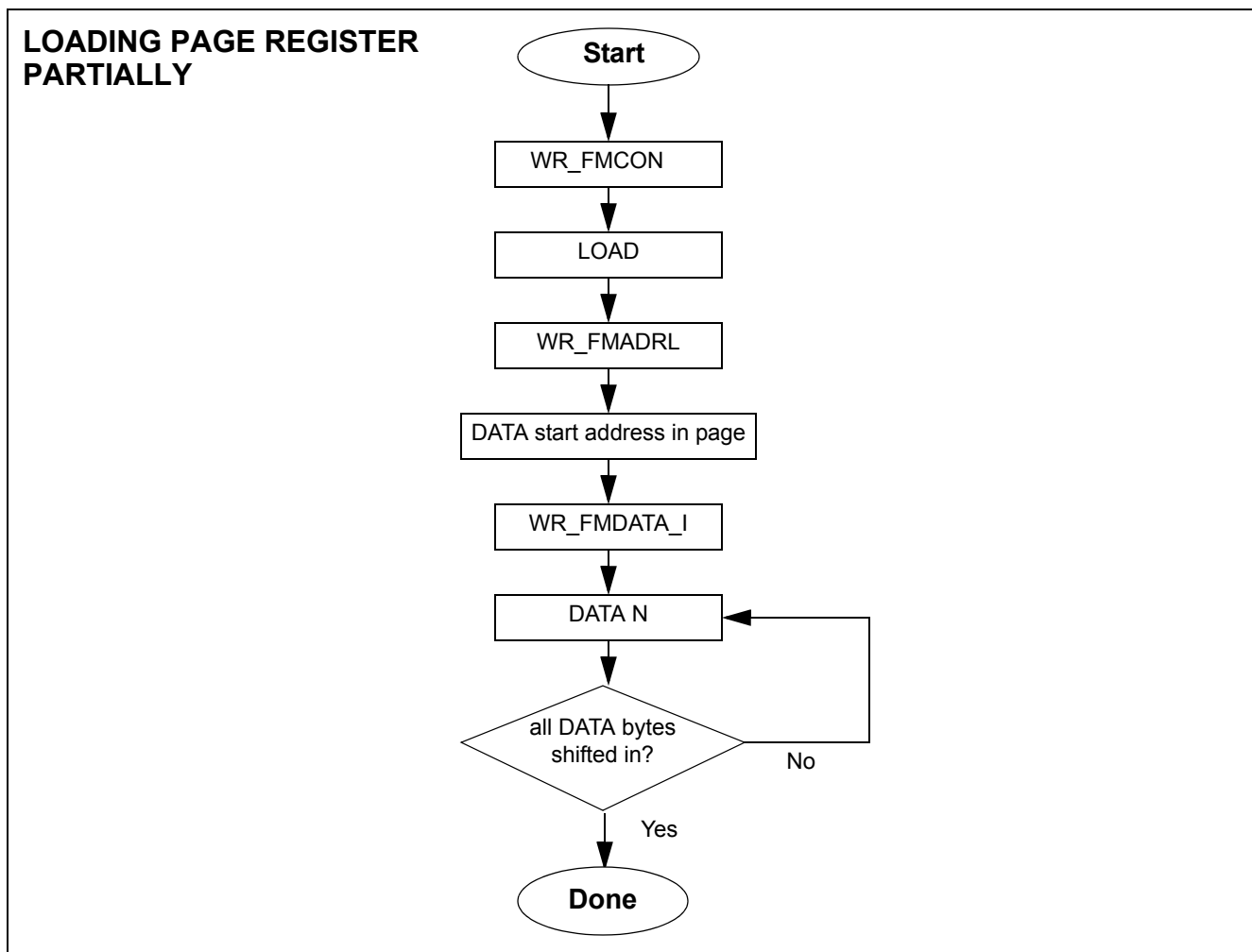
**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

**4.2 Loading the Page Register partially**

The page register is loaded by performing the following sequence:

1. Activate the ICP Programming Mode, as previously described, if not already performed.
2. Shift in the WR\_FMCON opcode.
3. Shift in 00h, the LOAD command to FMCON.
4. Shift in 08h, WR\_FMADRL opcode.
5. Shift in X0 to FMADRL. This sets the destination of the first data byte to be loaded as the beginning of the page.
6. Shift in the 14h, WRFMDATA\_PG opcode.
7. Shift in 16 bytes of data to be loaded into the page register. FMADRL is incremented after each byte. After the last byte FMADRL rolls over to point to the beginning of the page.

Example of a sequence loading the Page register when in programming mode:



**Figure 25: loading page register partially**

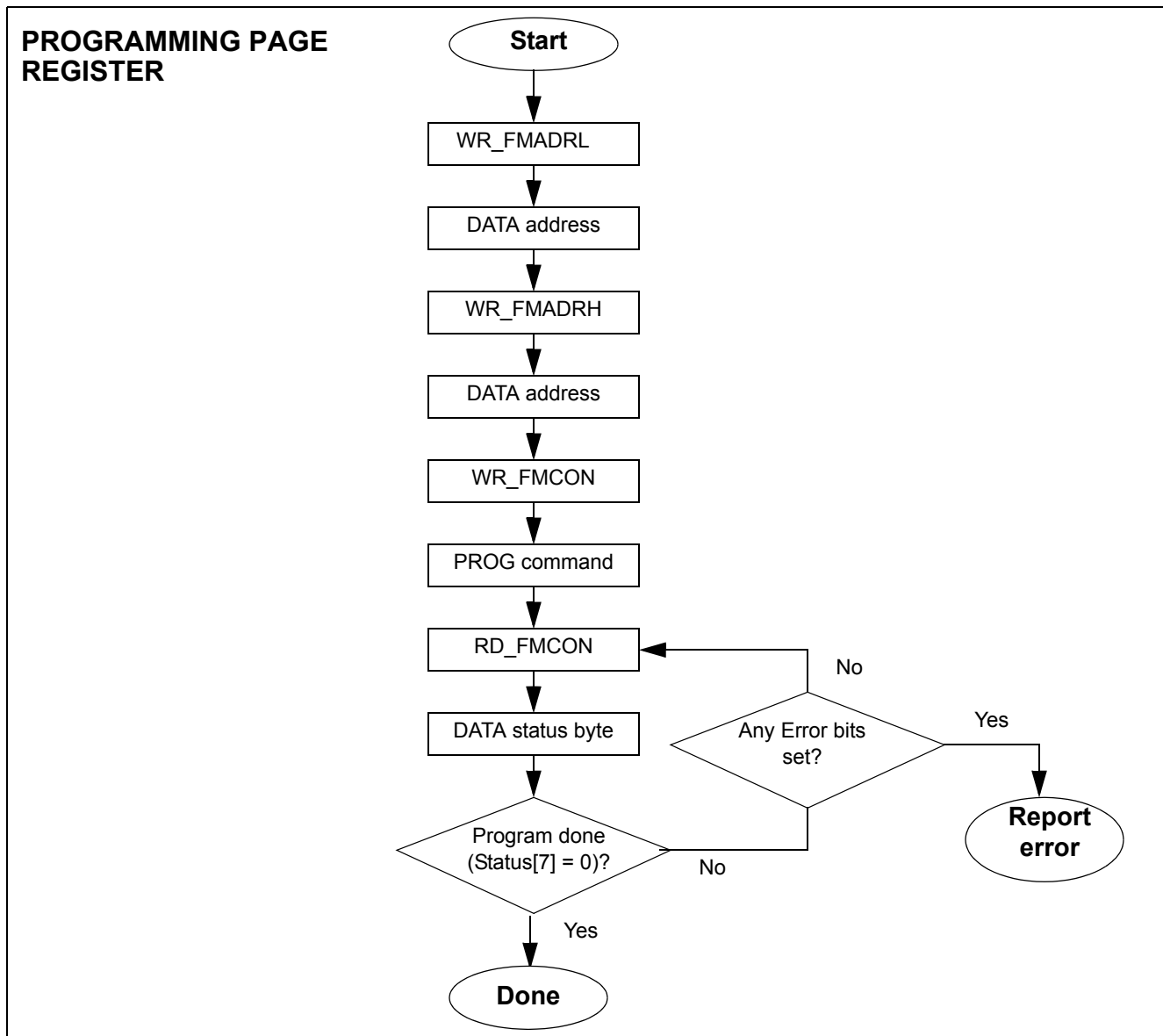
**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

**4.3 Programming User Code Memory**

Code memory may only be programmed by using the page register. This may be performed using the following sequence:

1. Load the page register with the data to be programmed as previously described.
2. Using the WR\_FMADRL opcode, write the lower 8-bits of the page address.
3. Using the WR\_FMADRH opcode, write the upper 8-bits of the page address.
4. Using the WR\_FMCON opcode, shift in the PROG command.
5. Read the FMCON register to obtain status. Continue reading until the interface is either not BUSY or until an error has occurred.

Example of a sequence programming the Page register in Flash when in programming mode:



**Figure 26: Programming page register**

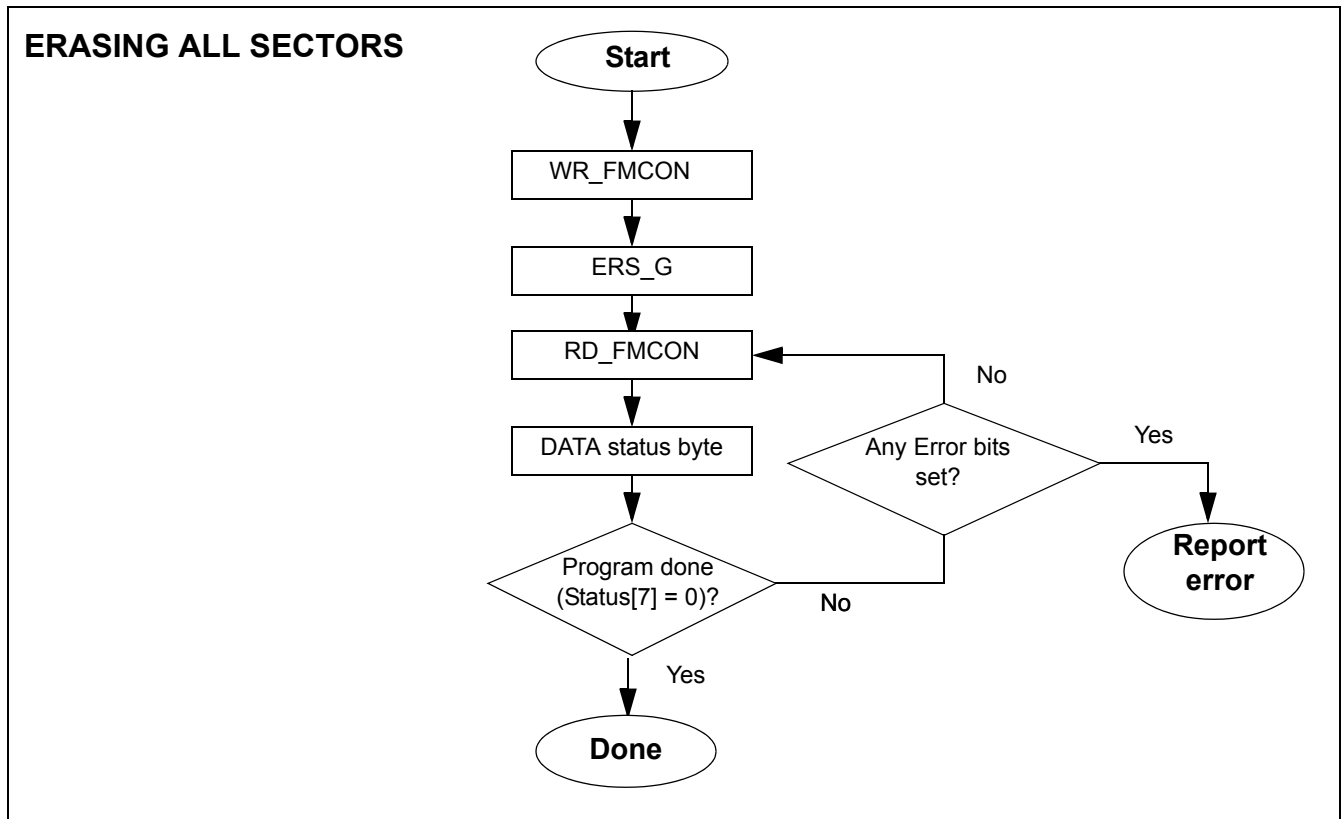
**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

**4.4 Erasing all sectors (global erase)**

Sectors and their sector security bits may be erased using the following sequence:

1. Shift in the WR\_FMCON command.
2. Shift in the ERS\_G command to the FMCON register
3. Shift in the RD\_FMCON command.
4. Continue reading until the interface is either not BUSY or until an error has occurred.

Example of a sequence erasing all sectors when in programming mode:



**Figure 27: Erasing all sectors**

**NOTE:** The Global erase has an errata. After writing ERS\_G read data using the FM\_DATA\_I command, after that RD\_FMCON till the busy flag gets cleared. If this is not done the busy flag will never get cleared. Please check erratasheets for which revisions this bug applies.

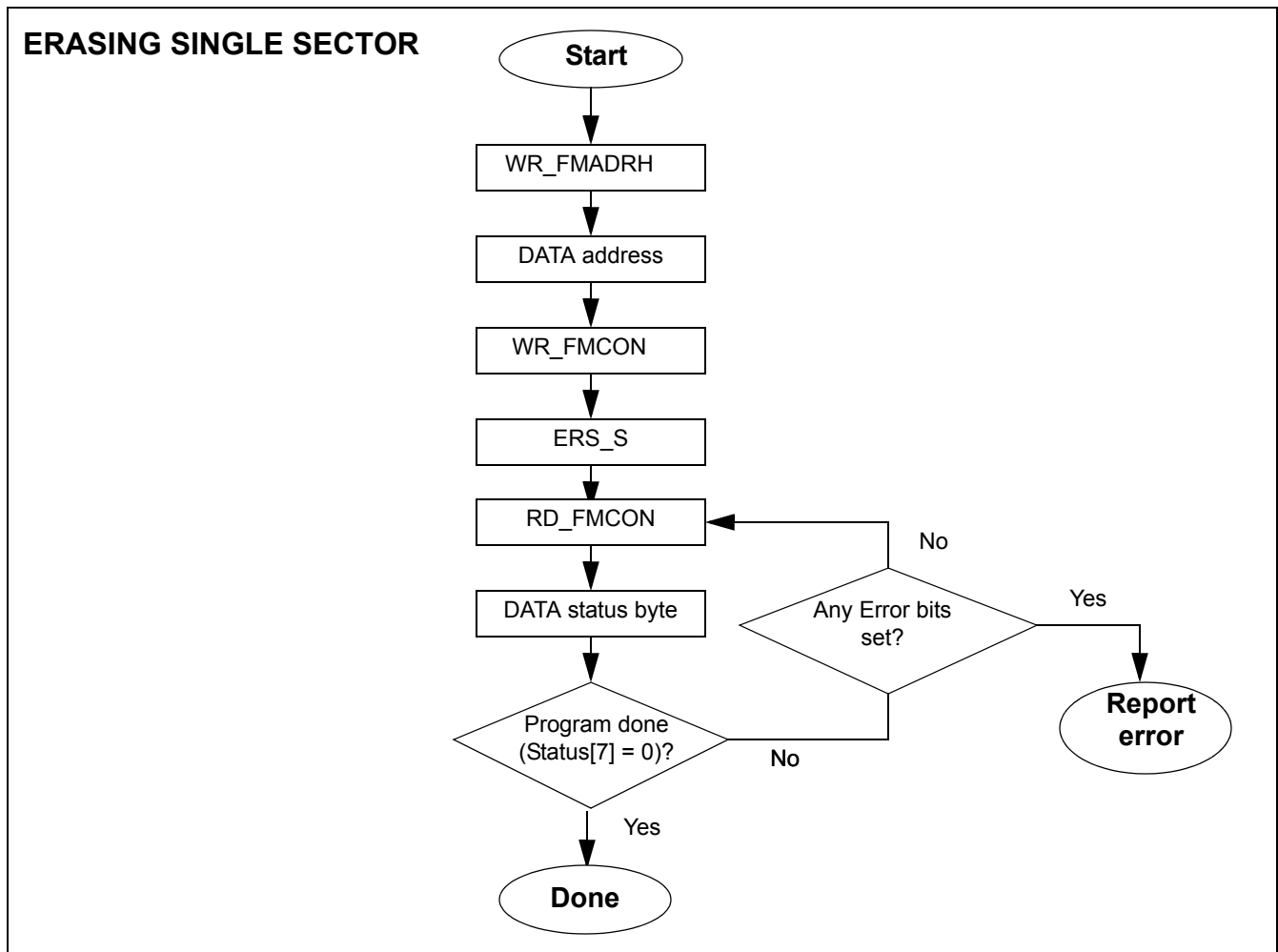
**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

**4.5 Erasing a single sector**

A single sector and it's sector security bits may be erased using the following sequence:

- 1.Shift in the WR\_FMADRH command
- 2.Shift in the upper address of the page
- 3.Shift in the WR\_FMCON command.
- 4.Shift in the ERS\_S command to the FMCON register
- 5.Shift in the RD\_FMCON command.
- 6.Continue reading until the interface is either not BUSY or until an error has occurred.

Example of a sequence erasing a single sector when in programming mode:



**Figure 28: Erasing single sector**

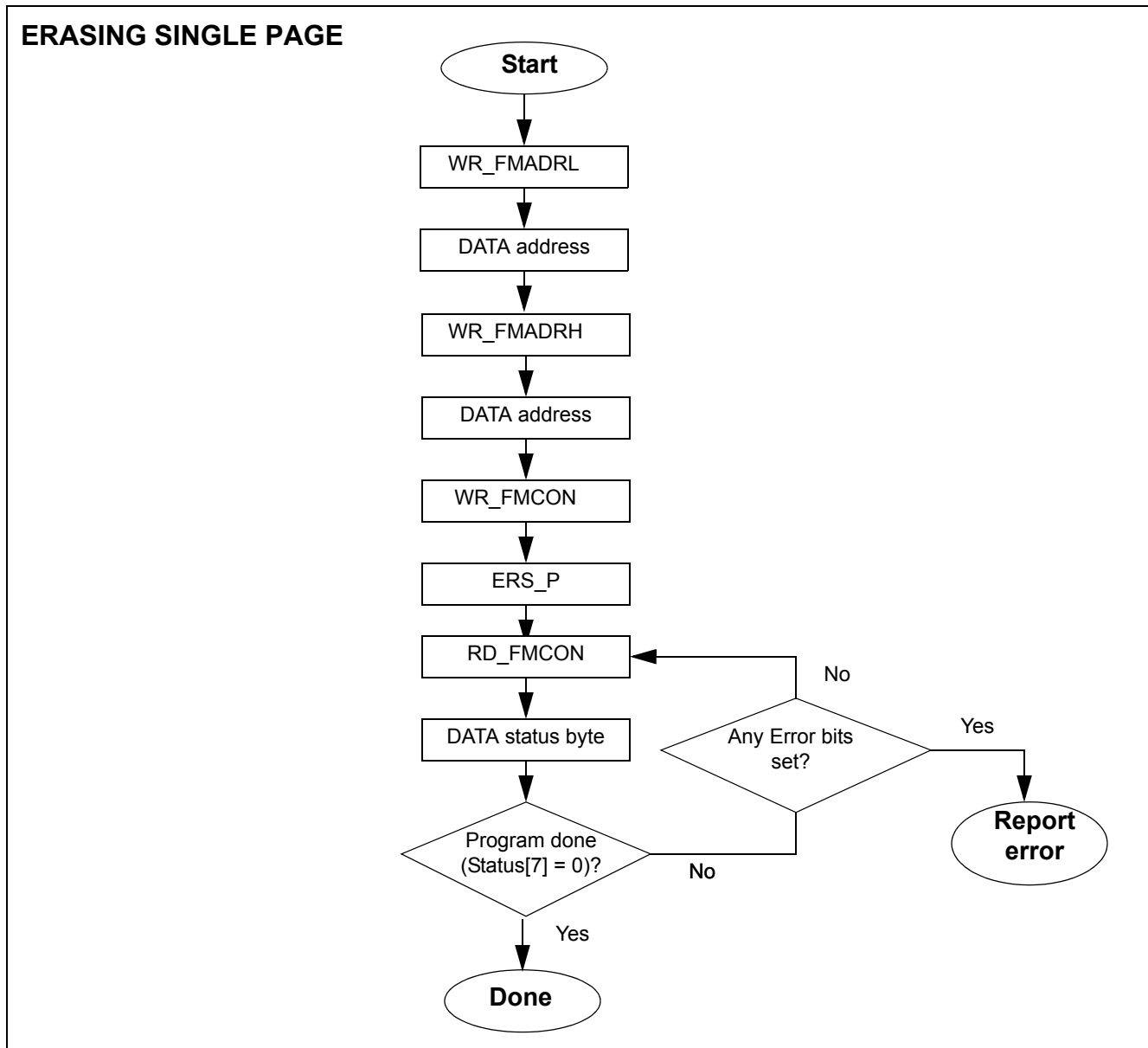
**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

**4.6 Erasing a single page**

A single page may be erased using the following sequence:

1. Shift the WR\_FMADRL command in.
2. Write the lower 8-bits of the page register address to FMADRL. (only FMADRL[7:6] are used)
3. Shift the WR\_FMADRH command in.
4. Write the upper 8-bits of the page register address to FMADRH.(only FMADRH[4:0] are used)
5. Write the ERS\_P command to the FMCON register.
6. Read the FMCON register to obtain status. Continue reading until the interface is either not BUSY or until an error has occurred.

Example of a sequence erasing a single page when in programming mode:



**Figure 29: Erasing single page**

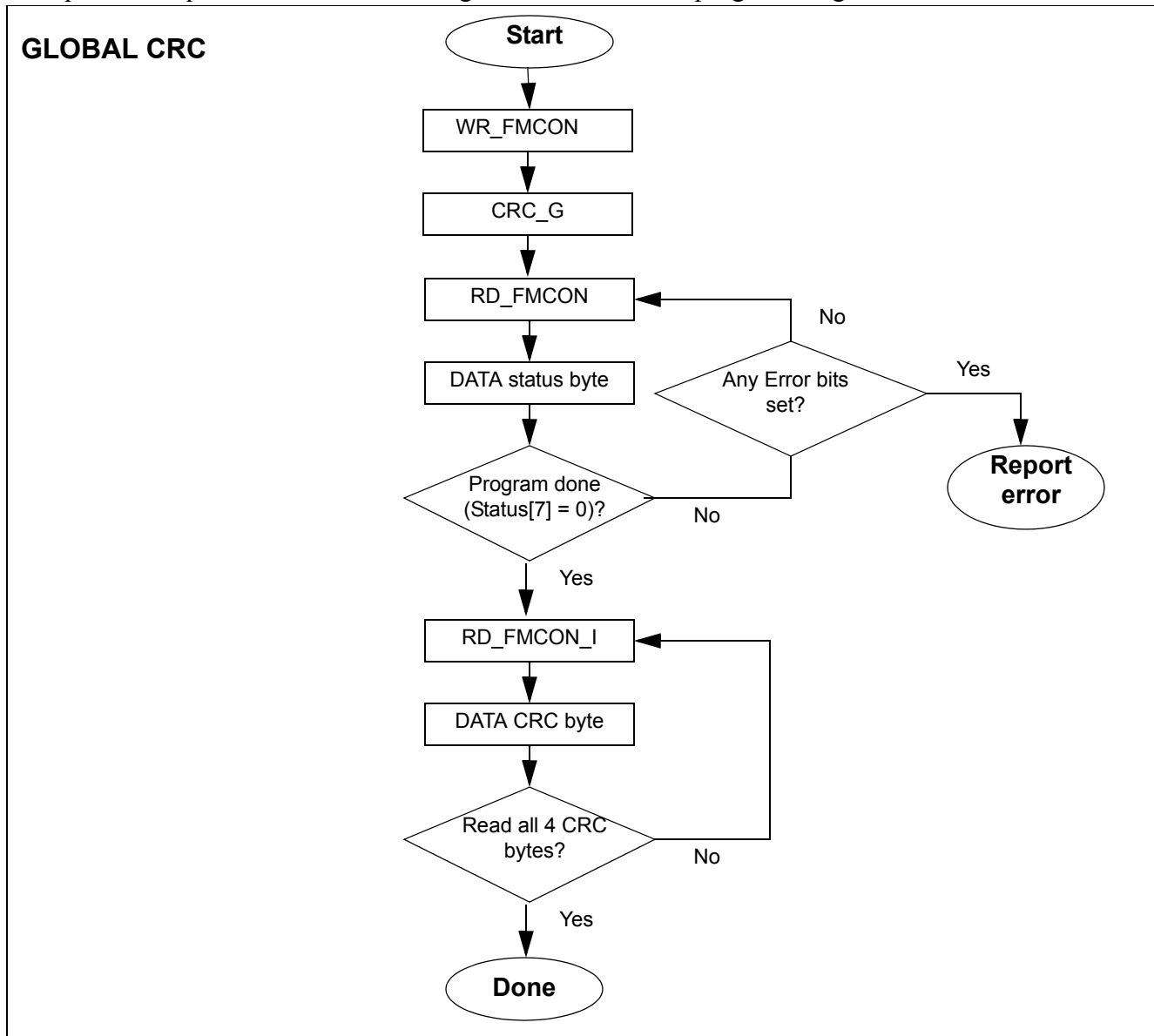
**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

**4.7 Calculate Global CRC**

A 32-bit global CRC of the entire user code memory may be calculated using the following sequence:

1. Shift in the WR\_FMCON command, followed by the CRC\_G opcode.
2. Shift in the RD\_FMCON command, keep reading FMCON till the interface is not BUSY or until it reports an error.
3. Shift in the RDFMCON command, and read out the 4 CRC bytes.

Example of a sequence that will read the global CRC when in programming mode:



**Figure 30: Global CRC**

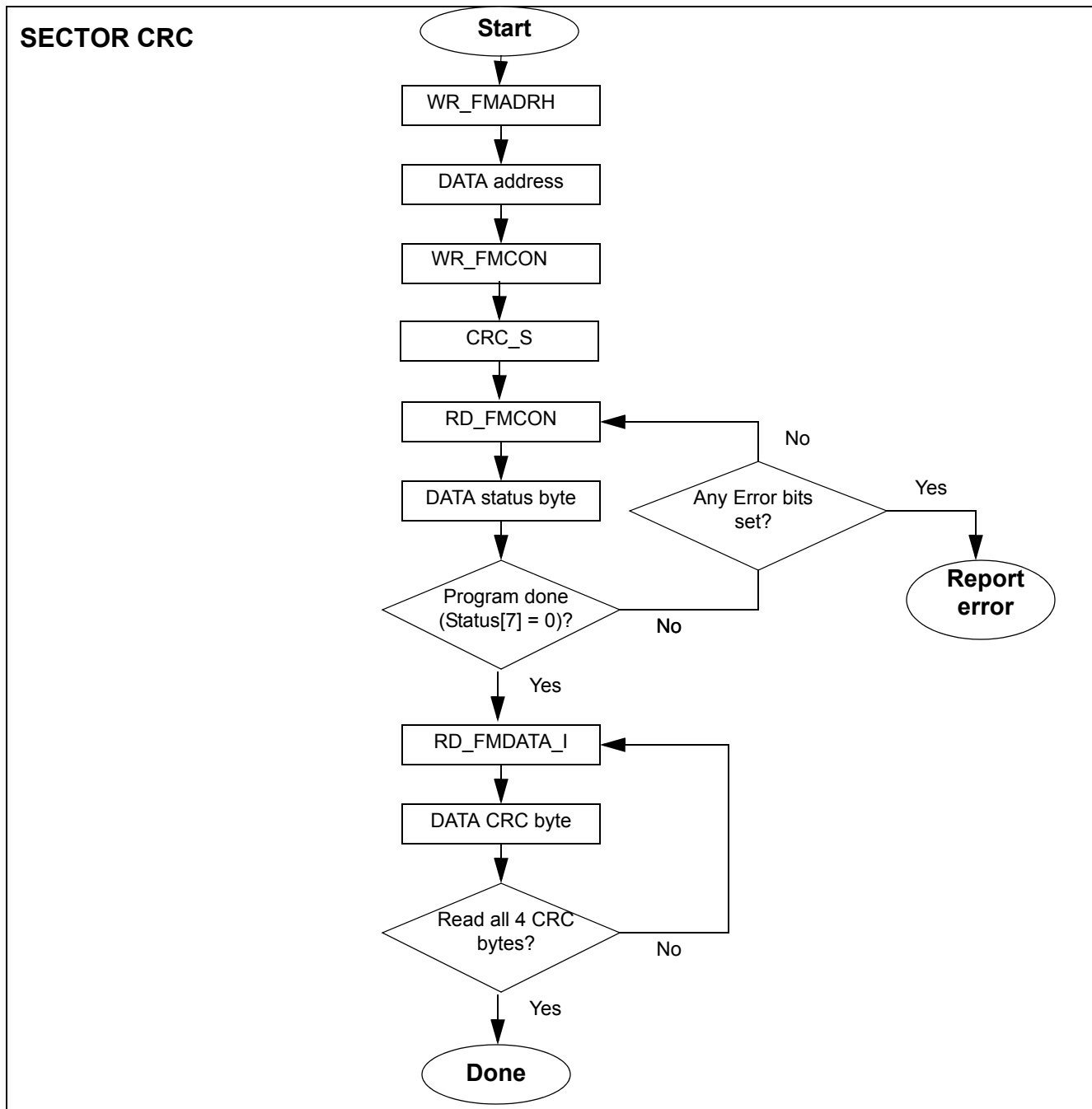
**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

**4.8 Calculate Sector CRC**

A 32-bit sector CRC of a single sector of user code memory may be calculated using the following sequence:

1. Shift in the WR\_FMCON
2. Shift in the WR\_FMCON command, followed by the CRC\_S opcode.
3. Shift in the RD\_FMCON command, keep reading FMCON till the interface is not BUSY or until it reports an error.
4. Shift in the RDFMCON command, and read out the 4 CRC bytes.

Example of a sequence that will read the CRC of a sector when in programming mode:



**Figure 31: Sector CRC**

**P89LPC9xx****In Circuit Programming (ICP) Specifications****4.9 Reading Configuration, Boot Vector, Status Byte, Security Bits, Signature Bytes**

Devices parameters such as configuration bytes, status byte, boot vector, security bits, and signature bytes may be read by writing an address of FMADRL and a command to FMCON. These registers have the following addresses:

Name	Mapped Address	Byte Description
UCFG1	00H	User Configuration Reg. #1 (UCFG1) program/read by the programmer
UCFG2	01H	User Configuration Reg. #2 (UCFG2) program/read by the programmer
Boot Vector	02H	Bootvector
Status Byte	03H	Status Byte
-	04H	-
-	05H	-
-	06H	-
-	07H	-
SEC0	08H	Security byte, sector 0
SEC1	09H	Security byte, sector 1
SEC2	0AH	Security byte, sector 2 <sup>2</sup>
SEC3	0BH	Security byte, sector 3 <sup>2</sup>
SEC4	0CH	Security byte, sector 4 <sup>1</sup>
SEC5	0DH	Security byte, sector 5 <sup>1</sup>
SEC6	0EH	Security byte, sector 6 <sup>1</sup>
SEC7	0FH	Security byte, sector 7 <sup>1</sup>
MFGID	10H	Manufacturer ID (READ ONLY)
ID1	11H	Device ID 1 (READ ONLY)
ID2	12H	Device ID 2 (READ ONLY)

**Table 5: EEPROM bytes**

**Notes:**

1. Applies only for the P89LPC922, P89LPC925, P89LPC931, P89LPC932A1, P89LPC934, LPC935
2. Applies for all parts except P89LPC920

## P89LPC9xx In Circuit Programming (ICP) Specifications

These bytes may be read using the following sequence:

- 1.Shift in the WR\_FMCON
- 2.Shift in the WR\_FMCON command, followed by the CONF opcode.
- 3.Shift in the WR\_FMADRL command.
- 4.Shift in the DATA address
- 5.Shift in the RD\_FMDATA command.
- 6.Shift out the data of the config byte.

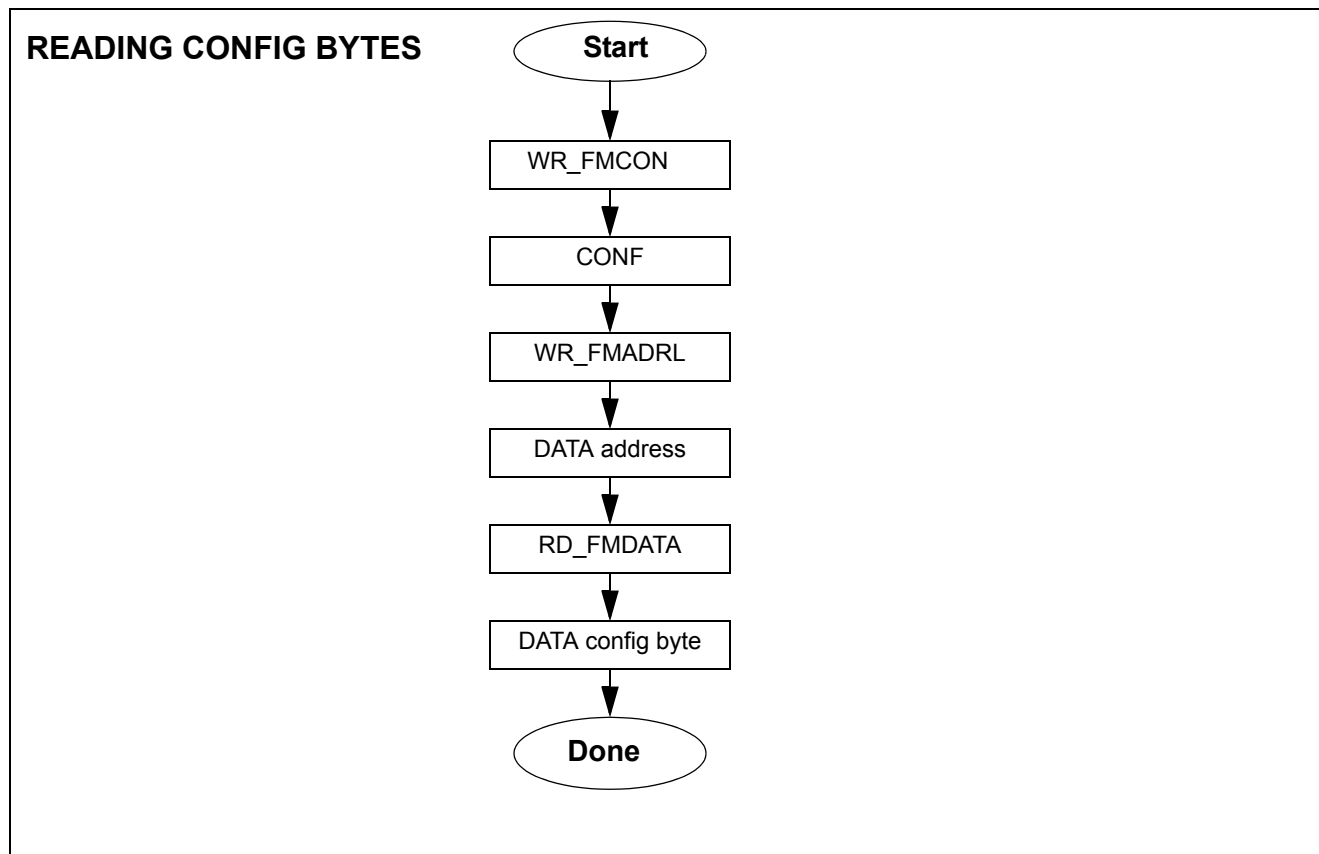


Figure 32: Reading config bytes

# P89LPC9xx

## In Circuit Programming (ICP) Specifications

### 4.10 Writing Configuration, Boot Vector, Status Byte, and Security Bits

Device parameters such as configuration bytes, status byte, boot vector, and security bits, made be written by writing a CONF command to FMCON, an address to FMADRL, and then the data to FMDATA.

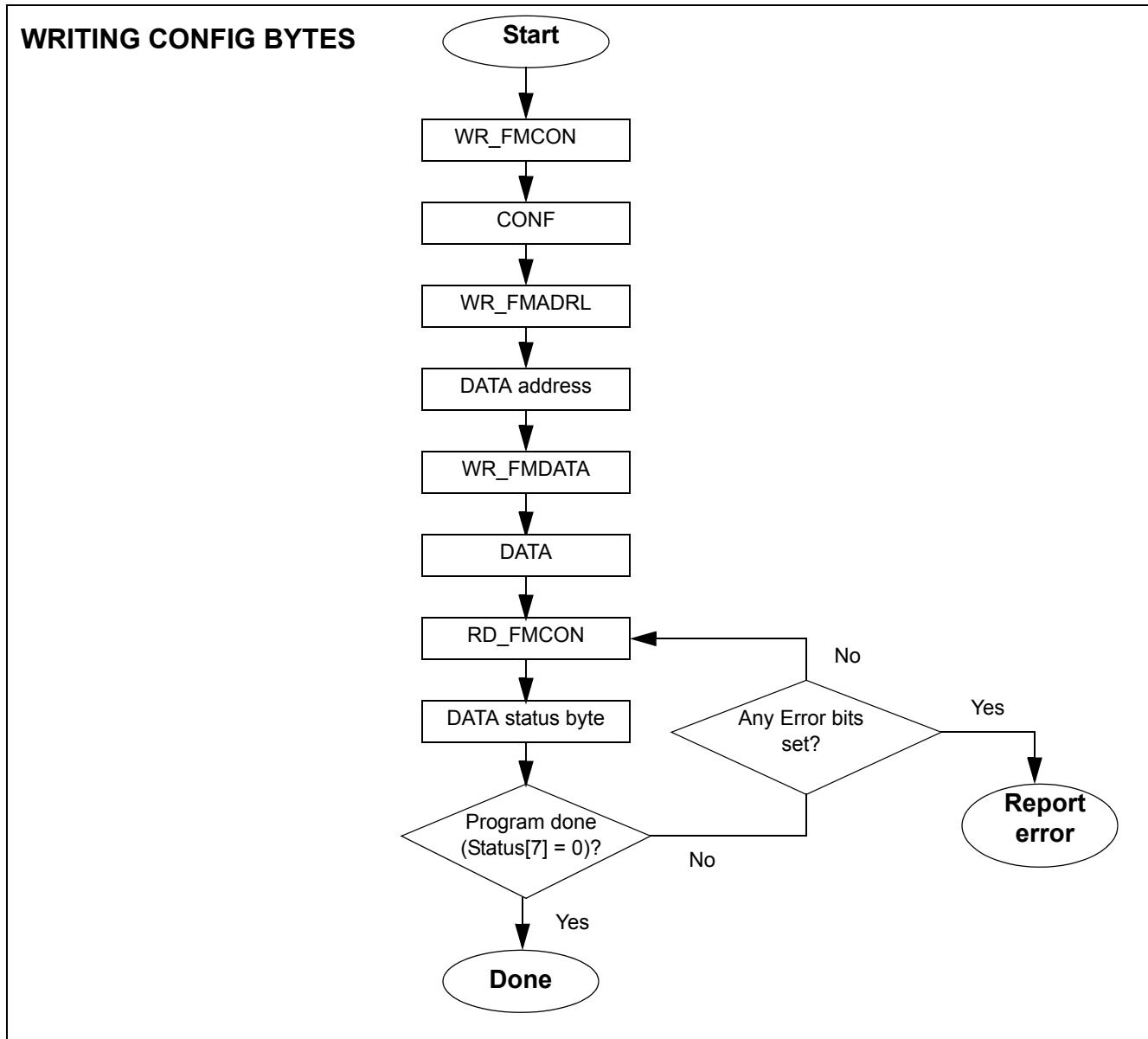


Figure 33: Writing config bytes

# P89LPC9xx

## In Circuit Programming (ICP) Specifications

### 4.11 Writing CCP Clear Configuration Protection

Configuration protection can be set in the Boot Status byte, to clear this protection the Clear Configuration Protection command has to be used.

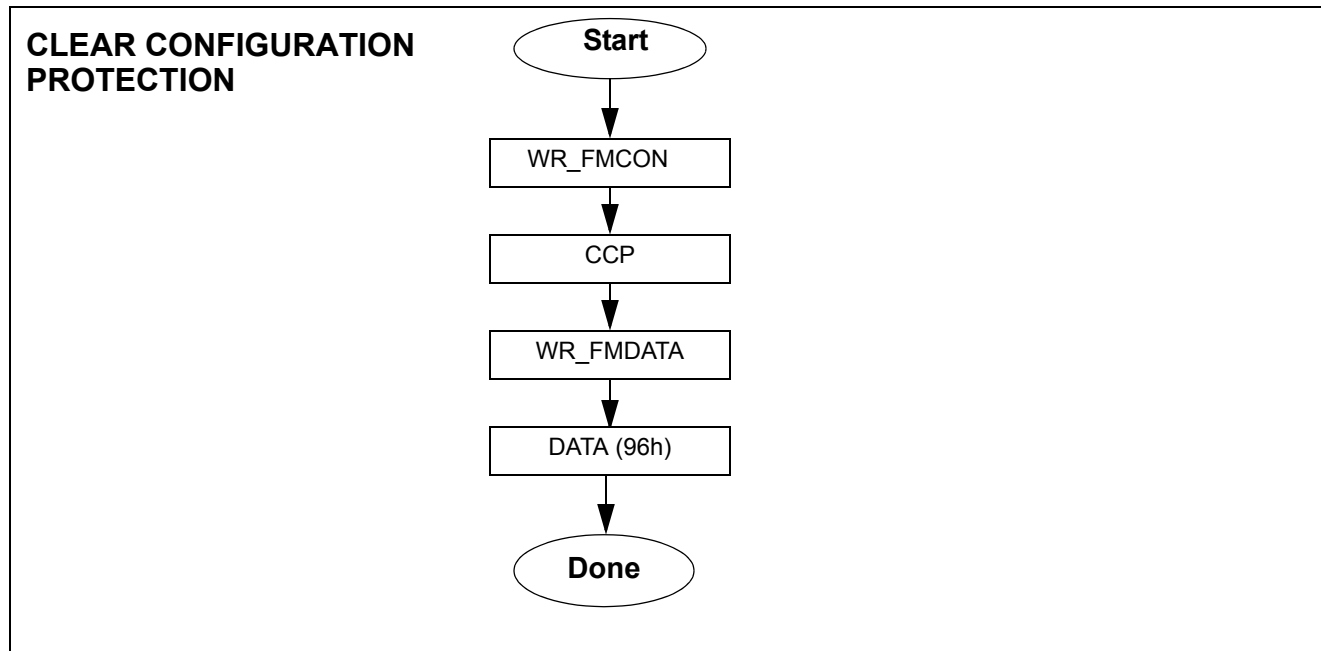


Figure 34: Clear Configuration Protection

**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

## 5.0 Calculating CRC

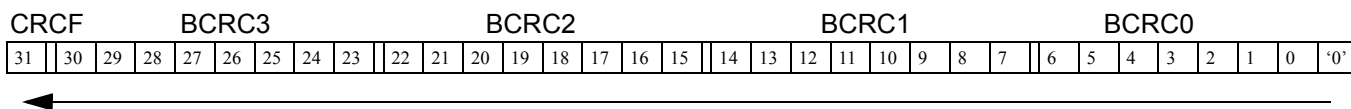
The method used to calculate a 32-bit CRC check for the LPC9xx, is described below. The calculation may be performed on either a sector or the entire user code space.

### INITIALIZATION

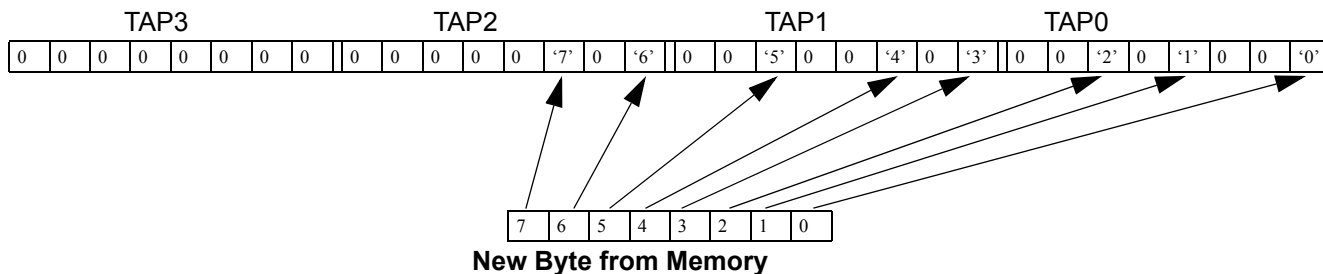
Define and initialize CRC polynomial, POLY0-POLY3 = 00400007H  
 Define and initialize a 32-bit CRC result, BCRC0-BCRC3 = 00000000H  
 Define and initialize a 32-bit temporary variable, TAP0-TAP3 = 00000000H  
 Define and initialize a 1-bit temporary variable to store the MSB of the CRC result, CRCF = 0.

STARTING WITH THE FIRST BYTE AND, FOR EACH BYTE IN THE MEMORY (either sector or entire array), PERFORM THE FOLLOWING CRC CALCULATION:

1. Shift the BCRC0-BCRC3 registers to the left by one and save the MSB in CRCF, this becomes the new CRC.



2. Get the byte from memory and perform the following operations on it.  
 a. Distribute the 8 bits of this new byte at the shown locations of the TAP registers and fill the rest of the TAP registers with 0s (direction of LSB to MSB is right to left).



b. XOR the corresponding TAP and BCRC registers and store the result back in the BCRC register.

BCRC0 = TAP0 XOR BCRC0  
 BCRC1 = TAP1 XOR BCRC1  
 BCRC2 = TAP2 XOR BCRC2  
 BCRC3 = TAP3 XOR BCRC3

c. If the CRCF bit was zero, the CRC calculation is done and BCRC3:0 holds the current CRC result. If the CRCF was a one, XOR the CRC polynomial and BCRC registers and store the new result back in the BCRC registers. BCRC3:0 hold the new 32-bit CRC.

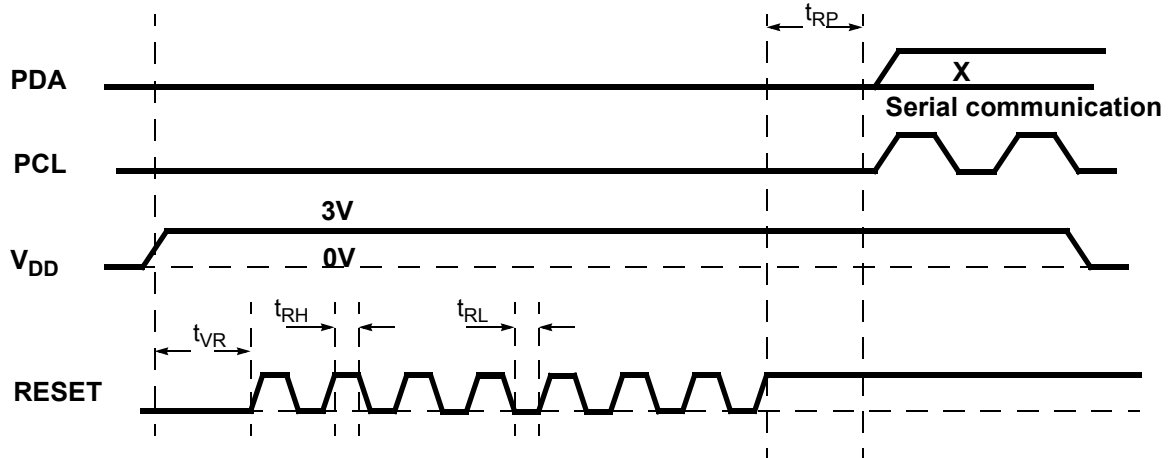
BCRC0 = POLY0 XOR BCRC0  
 BCRC1 = POLY1 XOR BCRC1  
 BCRC2 = POLY2 XOR BCRC2  
 BCRC3 = POLY3 XOR BCRC3

Repeat this operation, starting at step 1, above, for each byte in the memory array.

**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

## 6.0 AC timings

### 6.1 Getting into the serial programming mode



**Figure 35: Getting into the programming mode**

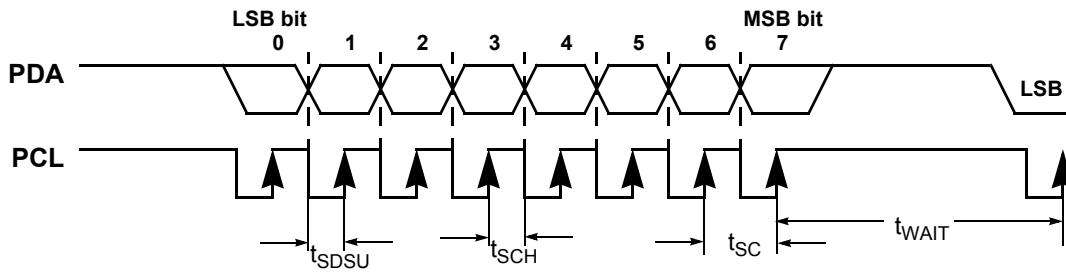
$V_{DD} = 2.4V$  to  $3.6V$ ;  $I_{PP} = TBD$  mA during programming;  $T_{amb} = 10^{\circ}C$  to  $+40^{\circ}C$

SYMBOL	FIGURE	PARAMETER	LIMITS		UNIT
			MIN	MAX	
$t_{VR}$		RST delay from $V_{DD}$ active	50	-	us
$t_{RH}$		RST HIGH time	1	32	us
$t_{RL}$		RST LOW time	1	-	us
$t_{RP}$		ICP entry time	TBD	TBD	us

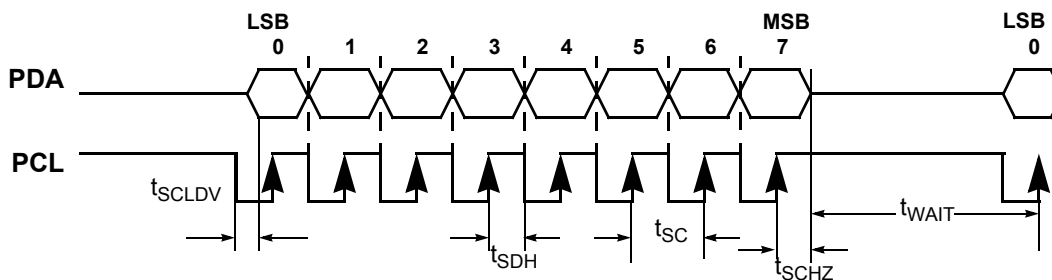
**Table 6: Programming mode timings**

**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

**6.2 Timing data shifting**



**Figure 36: Writing data**



**Figure 37: Reading data**

SYMBOL	PARAMETER	LIMITS		UNIT
		MIN	MAX	
$t_{SC}$	Serial clock cycle time	200	-	ns
$t_{SCH}$	Serial clock high	80	-	ns
$t_{SCL}$	Serial clock low	80	-	ns
$t_{WAIT}$	Wait between two serial bytes	200	-	ns
$t_{SDISU}$	Serial data input setup time to the rising edge of the serial clock	20	-	ns
$t_{SDIH}$	Serial data input hold time after the rising edge of the serial clock	40	-	ns
$t_{SDOH}$	Serial data output hold time after the rising edge of the serial clock	10	-	ns
$t_{SCLDV}$	Serial clock LOW to valid data of the first data bit in a byte	-	80	ns
$t_{SCHZ}$	Serial clock HIGH to data Hi-Z at the last data bit in a byte	-	80	ns

**Table 7: Programming commands timing**

## P89LPC9xx

### In Circuit Programming (ICP) Specifications

## 7.0 Revision History

<b>2005 September 26</b>	<b>Section</b>
Added P89LPC9221 ID byte	1
<b>2005 September 23</b>	<b>Section</b>
Added P89LPC9401	1
<b>2005 May 20</b>	<b>Section</b>
Added P89LPC952	1
<b>2005 January 19</b>	<b>Section</b>
Added Clear Configuration Protection	4
<b>2004 September 8</b>	<b>Section</b>
Added P89LPC9102/9103/9107/936/938	1
<b>2004 June 7</b>	<b>Section</b>
Added P89LPC932A1/904/915/916/917/933	1
<b>2003 September 8</b>	<b>Section</b>
Added CRC calculation	5
Added P89LPC920/930/931	1
<b>2003 July 31</b>	<b>Section</b>
Added P89LPC912/913/914/921/922/934/935	1
<b>2003 May 29</b>	<b>Section</b>
Added loading page register partially.	4
Added Global erase errata and workaround.	4
Deleted SEC4 - SEC7.	4
<b>2003 May 7</b>	<b>Section</b>
Fixed errors in reading CRC flowcharts and reading config flowchart	4
Corrected WR_FMDATA_PG and RD_FMDATA_PG commands	2
Added WR_FMDATA_I and RD_FMDATA_I commands for CRC reading	2
<b>2003 April 3</b>	<b>Section</b>
Added error messages in flowcharts	4
Updated shift timings	5
<b>2003 March 27</b>	<b>Section</b>
Added coversheet	-
Added table of contents	-
Added part types and packages	1
Added ID bytes	1
<b>2002 October 9</b>	<b>Section</b>
Added flow charts	-

---

**P89LPC9xx**  
**In Circuit Programming (ICP) Specifications**

---

<b>2002 September 10</b>	<b>Section</b>
Initial version	-

**Table 8: Revision History**