SIEMENS

Microcontrollers
ApNote AP0821

☐ additional file
APXXXX01.EXE available

C5xx / 80C5xx
In-System FLASH Programming

The following approach describes the proceeding for in-system reprogramming of an external (5V-only) FLASH code memory by using the internal ROM code. Due to the ´Havard Architecture´, an additional external logic (PLD) is used for a software switching mechanism between code and data memory.

K. Scheibert / Siemens HL MC AT

AP0821 ApNote - Revision History		
Actual Revision : Rel.01	Previous Revison: Rel. none	
Page of actual Rel.	Page of prev. Rel.	Subjects changes since last release)

In-System FLASH Programming with hardware implemented bank switching capability

The following information concerns all microcontrollers of the C5xx / SAB 80C5xx family which use an internal ROM mask in combination with an external code FLASH memory. The external FLASH memory can be used as reprogrammable code memory for the application software. This application focuses on SIEMENS 8-bit microcontroller derivatives with internal code memory (ROM) sizes of 8/16 Kbyte in maximum because of the overlapped code memory area of the FLASH memory of 8/16 Kbyte cannot be used:

- C501-1R
- C502-2R
- C504-2R
- C511(A)-R
- C513(A)-R/-2R
- C515-1R
- SAB 80C535
- SAB 80C537

This application note describes the proceeding for the in-system reprogramming of application software by using special application hardware for memory bank switching and software programming service routines in the internal mask programmable ROM for external FLASH memories (5V-only).

Due to the ´Harvard Architecture´ (separate code memory and data memory areas) erasing and writing of the FLASH memory only can be done when mapping it in the corresponding data memory area. This will be achieved by using a port pin as a software switch which distinguishes between ´Normal Mode´ and ´Programming Mode´ in combination with a PLD (Programmable Logic Device).

1 Memory Organization

In ´**Normal Mode**´, the 8/16 Kbyte of the application SW are located in the maskprogrammable internal ROM and cannot be modified in-system. Therefore the bank switching mechanism, erasing/programming/verifying software service routines for the external FLASH memory and user specific routines (which mustn´t be modified in the application) are located in the internal ROM. Interrupt service routines can be located in the external FLASH memory by programming corresponding LJMP instructions to interrupt vector addresses in the ROM location.

The hardware implemented ´code roll over mechanism´ in the C5xx/80C5xx derivatives when using internal ROM by EA# = ´1´ do switch on external bus accesses to the FLASH memory above the ROM boundary of 8/16 Kbyte. The PSEN# signal and the address and data information at P0/P2 will then be enabled automatically.

External accesses to the RAM located in data memory via MOVX instructions are possible in ´Normal Mode´ up to 64 Kbyte.

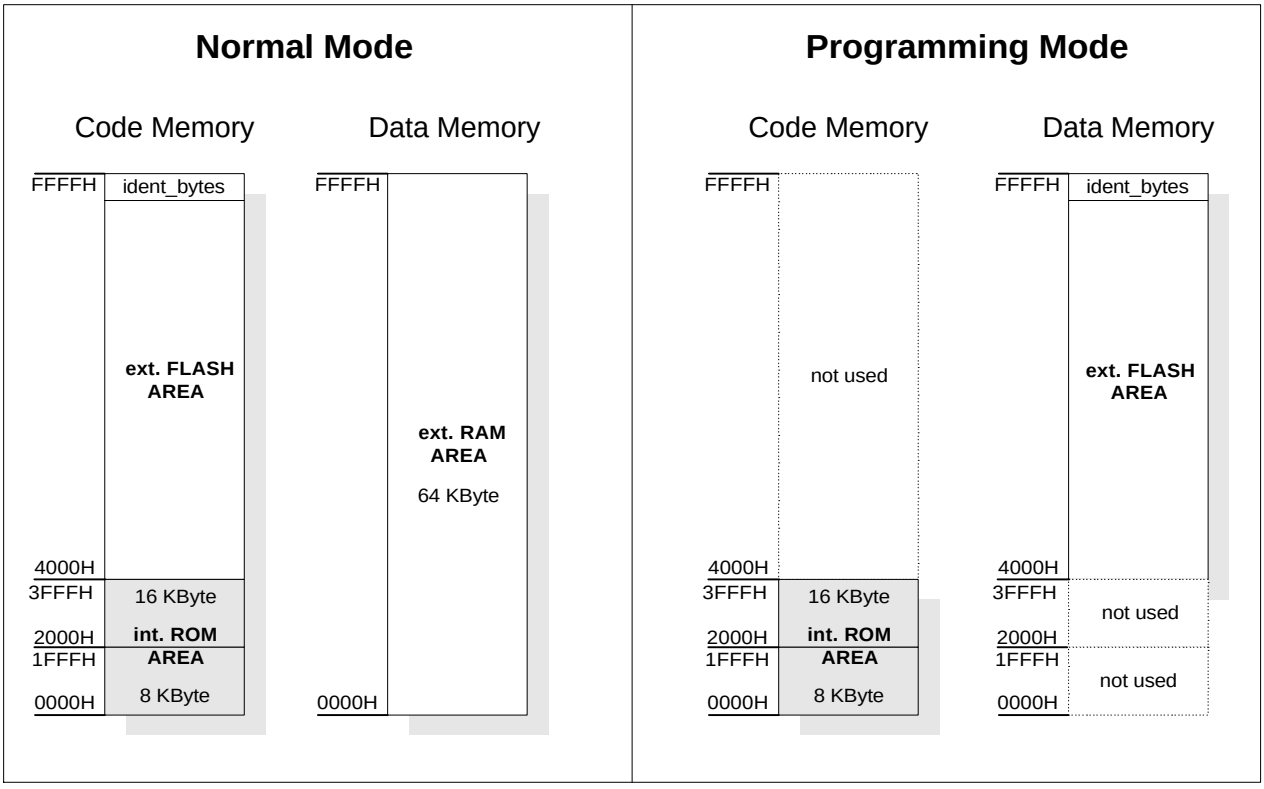

Figure 1:
Memory Organization for ´Normal Mode´ and ´Programming Mode´.

2 Hardware Description

In Figure 2 below, the circuit diagram of the bank switching architecture can be found:

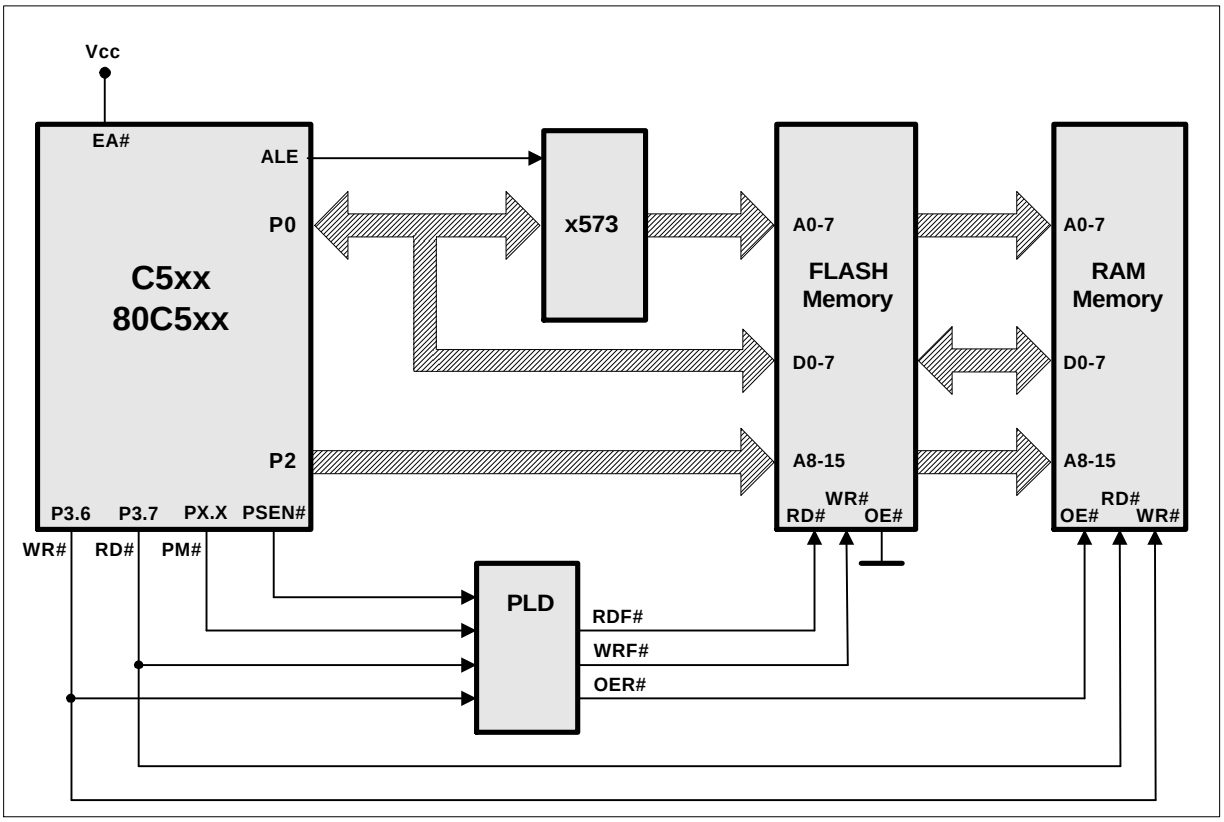

Figure 2:
Circuit Diagram with Bank Switching Mechanism

In Table 1 below, the signals used for the bank switching mechanism can be found:

Port Pin	Signal Name	Comment
PSEN#	PSEN#	**P**rogram **S**tore **EN**able signal is activated only at external bus accesses in the code memory area above 8/16 Kbyte
PX.X	PM#	**P**rogramming **M**ode signal is used for switching between ´Normal Mode´ and ´Programming Mode´
P3.7	RD#	**R**ea**D** signal is activated for external data memory read accesses in connection with MOVX instructions
P3.6	WR#	**WR**ite signal is activated for external data memory write accesses in connection with MOVX instructions
PLD.X	RDF#	**R**ea**D F**lash signal is connected to the RD# input of the FLASH memory
PLD.X	WRF#	**WR**ite **F**lash signal is connected to the WR# input of the FLASH memory
PLD.X	OER#	**O**utput **E**nable **R**AM signal is connected to the OE# input of the RAM

The PLD combines the output signals WR#, RD#, PSEN# and PM# to the output signals RDF#, WRF# and OER#. The signal PM# is used as a software switch for the bank switching algorithm. After power-on reset, this pin outputs ´1´and selects ´**Normal Mode**´ for program execution. In this mode the WRF# output of the PLD is disabled and PSEN# is wired to RDF# through the PLD. WR# and RD# are directly connected to the RAM and are used for accesses to the data memory area.

With setting PM# = ´0´, the bank switching mechanism switches over to ´**Programming Mode**´. In this mode the external RAM memory is disabled via OER# = ´1´ and the WR# and RD# input signals of the FLASH memory are wired to the corresponding WR# and RD# output signals from the microcontroller through the PLD. In this configuration it is possible to have external data accesses to the FLASH memory via MOVX instructions fetched from the internal code memory area (ROM). In this mode PSEN# remains at ´1´ because of internal code access in combination with EA# = ´1´. It is not permitted to execute code in the ´Programming Mode´ above the internal ROM size boundary of 8/16 Kbyte because of the inevitable activation of the PSEN# output.

The following equations for the PLD can also be found in Figure 3 below:

- RDF# = PSEN# AND (PM# OR RD#)
- WRF# = PM# OR WR#
- OER# = NOT PM#

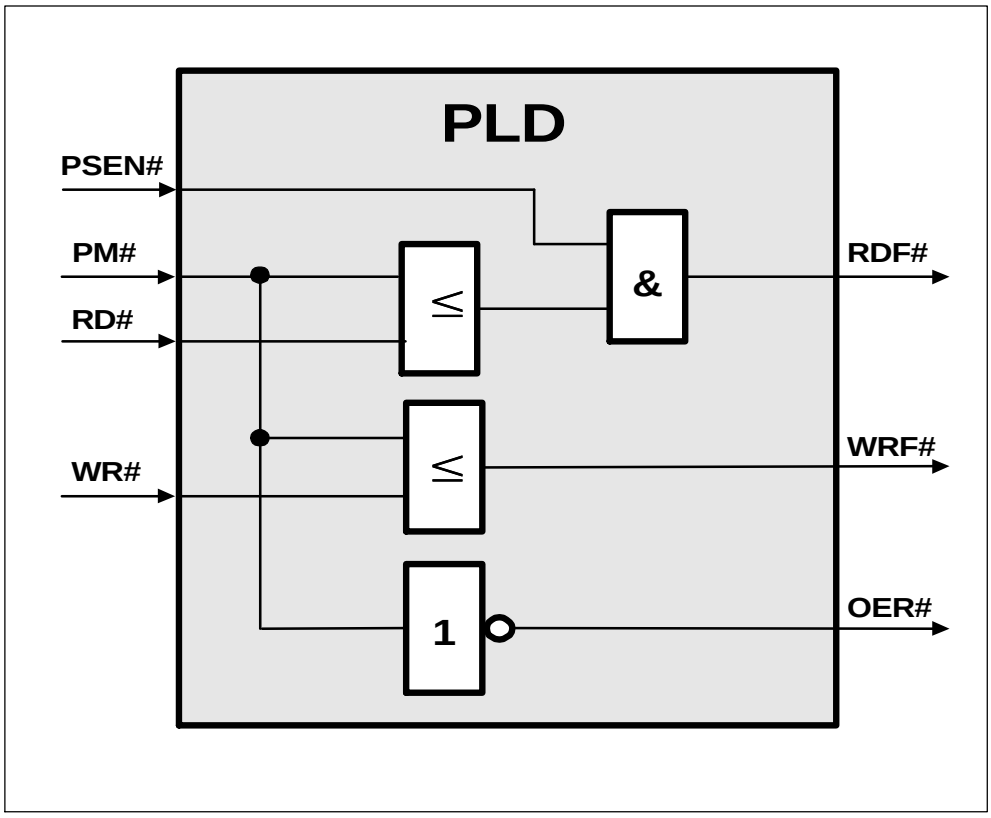

Figure 3:
PLD - Logical Equations

3 Functional Description of the ROM Software Routines

After power-on reset, the system automatically starts up with EA# = ´1´ in ´Normal Mode´ at address 0000H in the internal code memory area (ROM).

After the execution of the user specified initialization routines some ident_bytes (see Figure 1) in the upper FLASH memory location can be read out by using MOVC instructions. This ident_bytes are used as identifieres for a preprogrammed FLASH memory and should contain a useful combination of some bytes which normaly cannot occur in an emty device. This software check by the customer software can also contain a calculation of a checksum for safty purposes. If this software detects the FLASH memory as preprogrammed, the application software starts up regulary.

Otherwise the software will branch into the bank switching routine, PM# is set to ´0´ (selection of the ´Programming mode´) and the communication with an external host via serial interface has to be established. Source code then can be transfered from the external host to the FLASH memory by using erase/program and verify service routines, which are FLASH derivative specific.

After programming and verifying the new contents in the FLASH memory, PM# can be switched back to ´1´ = ´Normal Mode´ and the application software is able to start up now.

Some customers may have the demand for reprogramming their application system e.g. for in-system software updates. In this case a special diagnostics mode has to be implemented in the internal ROM location, which can be detected e.g. by using a special diagnostics input pin. When starting up, the diagnostic input pin will be tested and software branches to the diagnostic routines, which allows the reprogramming of the external FLASH memory.

The proposed solution represents an approach towards in-system programming using FLASH technology implemented on Siemens microcontrollers. As these devices can easily be programmed for several times, this structure leads to reduced system costs and a comfortable way of changing program code.