

UCD30xx

Address Manager (DEC)

Programmer's Manual

Literature Number: xxxxxx

8/6/2008

Table of Contents

UCD30xx	1
Address Manager (DEC).....	1
Programmer's Manual	1
Literature Number: xxxxxx.....	2
8/6/2008	2
1 DEC Address Manager Summary.....	3
2 Memory Mapping Basics.....	3
2.1 Why change the Memory Map?.....	3
2.2 Why are the Memory Map Registers so complex?.....	3
2.3 How do the Memory Map Registers work?.....	3
3 Boot ROM memory initialization	3
4 Erasing and Programming Flash.....	3
4.1 Waiting for flash operations to finish	3
4.2 Erasing Data Flash	3
4.3 Writing to Data Flash.....	3
4.4 Erasing Program Flash.....	3
4.5 Writing to Program Flash.....	3
5 DEC – Address Manager Reference	3
5.1 Memory Fine Base Address High Register 0 (MFBHR0).....	3
5.2 Memory Fine Base Address Low Register 0 (MFBALR0).....	3
5.3 Memory Fine Base Address High Register 1-5 (MFBHRx).....	3
5.4 Memory Fine Base Address Low Register 1-5 (MFBALRx).....	3
5.5 Program Flash Control Register (PFLASHCTRL).....	3
5.6 Data Flash Control Register (DFLASHCTRL)	3
5.7 Program Flash Interlock Register (PFLASHILOCK).....	3

1 DEC Address Manager Summary

The DEC Address Manager controls memory mapping and flash programming. All memory mapping activity is normally done by the boot ROM in the 3040. This is described in this document.

This information is only useful if there is some need to return to ROM mode without going through a reset first.

There may be a need for customer programs to erase and program both Program and Data Flash, so this is discussed in some detail.

2 Memory Mapping Basics

There are 4 address spaces in the UCD30xx. Each memory space has a pair of registers to set its address and block size:

Register Number	Memory	Size in bytes	Address at reset	Address in ROM Mode	Address in Flash Mode
0	Boot ROM	4K	Most of space	0 – ffff	a000 - afff
1	Program Flash	32K	-----	10000 – 17fff	0 – 7fff
2	Data Flash	2K	-----	18800 – 18fff	
3	RAM	4K	-----	19000 – 19fff	

Here is a figure:

	Reset Mode	ROM Mode	Flash Mode	Vectors
0	ROM	ROM		
0x1000	ROM	ROM		
0x2000	ROM	ROM		
0x3000	ROM	ROM		
0x4000	ROM	ROM		
0x5000	ROM	ROM		
0x6000	ROM	ROM		
0x7000	ROM	ROM		
0x8000	ROM	ROM		
0x9000	ROM	ROM		
0xA000	ROM	ROM		
0xB000	ROM	ROM	ROM	
0xC000	ROM	ROM		
0xD000	ROM	ROM		
0xE000	ROM	ROM		
0xF000	ROM	ROM		
0x10000	ROM			
0x11000	ROM			
0x12000	ROM			
0x13000	ROM			
0x14000	ROM			
0x15000	ROM			
0x16000	ROM			
0x17000	ROM			
0x18000	ROM			
0x19000	ROM	Data FLASH	Data FLASH	
0x1A000	ROM	RAM	RAM	

Program
FLASH

ROM

ROM always
executes at
0xA000
(except for
Vectors)

Program
FLASH

Data FLASH

RAM

Data FLASH

RAM

2.1 Why change the Memory Map?

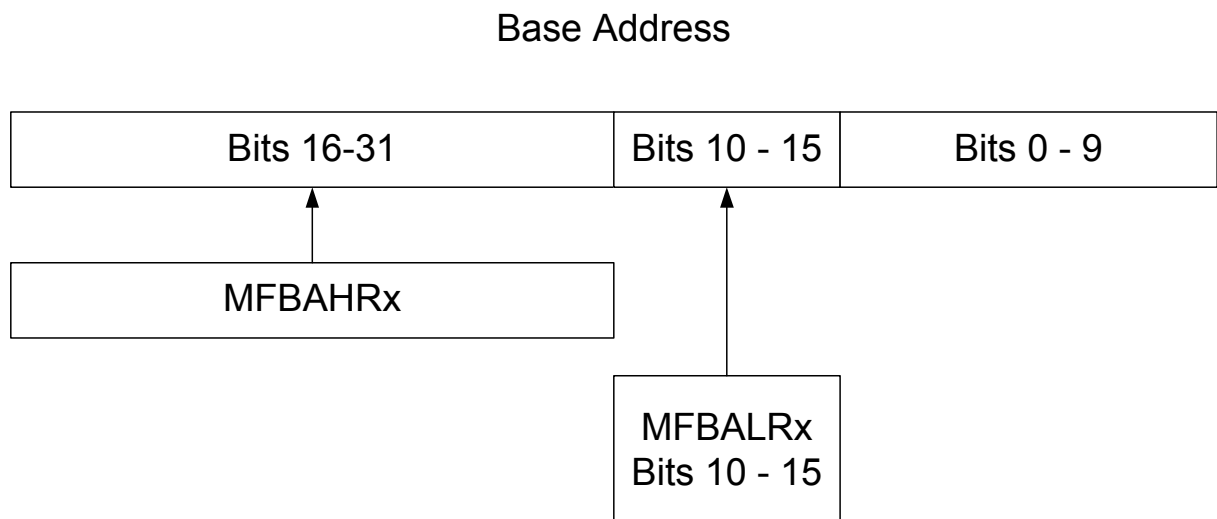
The memory map changes primarily because the vectors – for reset, interrupts, and faults – are located starting at location 0 in memory. When the UCD30xx powers up, the ROM needs to control the vectors. When the memory is configured, but the ROM is still executing, the ROM still needs to control the vectors. Then when control is handed over to the customer program in FLASH, the vectors need to be assigned by the customer for most efficient execution.

2.2 Why are the Memory Map Registers so complex?

The ARM core, as well as the Memory Map control, were designed for complex systems with external memory – personal computers, for example. It was more cost and time effective to reuse the memory map block from a more complex system than to devise a new one.

2.3 How do the Memory Map Registers work?

Two registers, the Memory Fine Base Address High Register, and the Memory Fine Base Address Low Register, combine to provide a start address for the memory block.



The Block Size field in the Memory Fine Base Address Register Low determines the size of the area for the memory block. The possible sizes range from 1K bytes, represented by a 1 in BLOCK_SIZE, to 16M bytes, represented by a 0xF. A zero in BLOCK_SIZE turns that particular memory select off.

To calculate the address provided by the MBBALRx, multiply it by 4 and put it in bits 8 to 15. For example, a 0x22 translates to 0x88000. A 0x24 translates to a 0x9000.

There are other bits as well, for example an RONLY bit signifying that the memory is read only. There is also a MS (for Memory Select) bit in only MFBALR0, which enables the entire DEC module.

DEC stands for Memory Address **DEC**ode.

3 Boot ROM memory initialization

When the UCD3040 powers up, the ROM is the only memory in the entire memory space. The peripherals are also present at the extreme high end of the memory space. The ROM image is repeated throughout the rest of the memory. The reset vector at location 0 in memory is accessed by the first instruction fetch of the CPU.

This instruction is a branch to the ROM entry point, aimed at the ROM image starting at address 0xa000. This is done because eventually the ROM will be mapped to 0xa000 to 0xa000 only. If the program counter is already pointing there, there is no discontinuity when the address is changed.

First in the ROM program, the three address register pairs for the other memories are changed.

Memory Fine Base Address High Registers 1-3 (MFBABRx) are all loaded with a 1. This register provides bits 16 – 31 of the address for the device, so all of the memories will be mapped to 0x1xxxx.

The Memory Fine Base Address Low Registers 1 – 3 (MFBALRx) are loaded as follows:

MFBALR1 – The BLOCK_SIZE field is loaded with a 6. This means that the Program Flash is at 0x10000, and is 32K bytes long.

MFBALR2 – BLOCK_SIZE = 2, ADDRESS = 0x22 – Data flash = 2K bytes at 0x18800

MFBALR3 – BLOCK_SIZE = 3, ADDRESS = 0x24 – RAM = 4K bytes at 0x19000.

Now that the other memories are mapped, the ROM is moved to fill the entire 64K space starting at zero. This is done at this point so that the ROM can still control the interrupt vectors at 0, all the other memories can be read and written, and the ROM can still execute at 0xa000.

This is done by modifying the MFBALR0 register, which controls the ROM address, and also enables the memory map.

MFBALR0 – BLOCK_SIZE = 7, RONLY = 1, MS = 1. (ADDRESS is left at 0)

This does all that is described above, as well as making it a fault if the program attempts to write to the ROM. All of the statements above are shorthand descriptions, not C code. In fact, to save space in the ROM a single constant is written to each register, with all the bit fields properly placed in it.

After the memory map is initialized in this pattern, the ROM program performs a simple additive checksum on the Program Flash. If the checksum matches, the ROM program then reconfigures the memory map and jumps to location zero in the flash.

There is also a PMBus command that can command the ROM program to do the same thing.

This reconfiguration involves two steps:

1. Remap the ROM to 0xa000 only
2. Remap the Program Flash to location 0

So the memory map when the Program Flash is running is:

Program Flash

Memory	Start	End
Program Flash	0	0x7fff
ROM	0xa000	0xa000
Data flash	0x18800	0x18fff

Program Flash	0x19000	0x19fff
---------------	---------	---------

There should be no need to modify any of the memory base address registers, so no further documentation is provided. Modifying them is dangerous. It is very easy to cause a fault which will cause the CPU to be reset.

4 Erasing and Programming Flash

There will be frequent need for programming data Flash memory, for changing default values, for calibration, and for data logging. Program flash modification is much more difficult, and less likely to be needed.

Note that all flash operations other than read involve considerable delay. Erase delays, especially, can be several milliseconds. Consult the datasheet for specific delay information for the device of interest.

4.1 *Waiting for flash operations to finish*

All flash modifications take more than one instruction cycle to complete. Both flash control registers have a BUSY bit that should be checked to verify that no flash process is already underway.

4.2 *Erasing Data Flash*

Data flash and Program Flash have separate flash programming circuitry, so it is possible to operate on data flash while executing from program flash. It is not possible to read from data flash while writing or erasing data flash, however. So all values that will be needed during the erase/write process should be stored in RAM or program Flash.

The UCD3040 has 2048 bytes of Data flash organized in 64 blocks of 32 bytes each. Erasing can be done a block at a time. To erase a block, simply write to the Data Flash Control Register (DFLASHCTRL) with the block number in the low 6 bits (PAGE_SEL) and a 1 in bit 9 (PAGE_ERASE).

Then wait for the BUSY bit in the same register to go low before doing any other Data Flash Operation.

To erase the entire Data Flash at once, simply write to the same register with bit 8 (MASS_ERASE) set.

Erasing Data Flash (or Program Flash) sets all the bits in the Flash locations.

4.3 *Writing to Data Flash*

Writing to Data Flash is very simple. Just write to the location. It is then necessary to monitor the BUSY bit in order to determine when the write is done.

On many flashes, it is possible to write multiple times to the same location without an erase, so long as more bits are being cleared. On the Data Flash on the UCD30xx family this is not the case. There is additional correction logic and additional flash bits to permit correction of a single bit error in Data Flash. Trying multiple writes to the same location without an erase in between will have unpredictable results because of this.

4.4 Erasing Program Flash

Erasing Program Flash is exactly the same as the procedure for Data Flash, above, except that Program Flash is divided into 32 pages of 1024 bytes each.

The other issue with Erasing Program Flash is that this is the normal location for programs to run, and it is not possible to execute from Program Flash while erasing it or writing to it. So a suitable program must be placed in either Data Flash or RAM, and executed while the functions are being performed.

4.5 Writing to Program Flash

There are two issues making Program Flash writes more complex than Data Flash writes:

1. The program must execute from Data Flash or RAM for the entire time of the write
2. It is necessary to write a key to the Program Flash Interlock Register (PFLASHILOCK) register before each write to the Program Flash. The key is given in the Reference Section below.

Other than that, the process is exactly the same as the process for writing to Data Flash – simply write to the actual address, and then monitor the BUSY bit.

5 DEC – Address Manager Reference

The DEC generates the memory selects and SAR peripheral select signals by decoding the address and control signals from the ARM processor. In addition, the DEC provides the control signals for the Program and Data Flash.

The assigned memory selects for the 3040 are as follows:

Memory Select 0 => Boot ROM (1Kx32)
Memory Select 1 => Program Flash (8Kx32)
Memory Select 2 => Data Flash (512x32)
Memory Select 3 => Data RAM (1Kx32)

5.1 Memory Fine Base Address High Register 0 (MFBAHR0)

Address FFFFE00

Bit Number	15:0
Bit Name	ADDRESS[31:16]
Access	R/W
Default	0000_0000_0000_0000

Bits 15-0: ADDRESS[31:16] – 16 Most Significant Bits of the Base Address. The Base Address sets the 22 most significant bits of the memory address.

5.2 Memory Fine Base Address Low Register 0 (MFBALR0)

Address FFFFE04

Bit Number	15:10	8	7:4	1	0
Bit Name	ADDRESS[15:10]	MS	BLOCK_SIZE	RONLY	PRIV
Access	R/W	R/W	R/W	R/W	R/W
Default	000000	0	0000	0	0

Bits 15-10: ADDRESS[15:10] – 6 Least Significant Bits of the Base Address. The Base Address sets the 22 most significant bits of the memory address.

Bit 8: MS – Memory Map Select

0 = Memory Map configuration not updated (Default)

1 = Enables the fine and coarse memory selects and activates the memory map

Bits 7-4: BLOCK_SIZE – Configures the size of the memory

0000 = Memory select is disabled (Default)

0001 = 1K Bytes

0010 = 2K Bytes

0011 = 4K Bytes

0100 = 8K Bytes

0101 = 16K Bytes

0110 = 32K Bytes

0111 = 64K Bytes

1000 = 128K Bytes

1001 = 256K Bytes

1010 = 512K Bytes

1011 = 1M Bytes

1100 = 2M Bytes

1101 = 4M Bytes

1110 = 8M Bytes

1111 = 16M Bytes

Bit 1: RONLY – Read-only protection. This bit sets read-only protection for the memory selected by the memory select. An illegal access exception is generated when a write is attempted to the memory.

0 = Read/write access to memory (Default)

1 = Read accesses to memory only

Bit 0: PRIV – Privilege mode protection. This bit sets privilege mode protection for the memory Registration selected by the memory select. An illegal access exception is generated on any access to memory protected by privilege mode.

0 = User/privilege mode accesses to memory (Default)

1 = Privilege mode accesses to memory only

5.3 Memory Fine Base Address High Register 1-5 (MFBAHRx)

Address FFFFFFFE08 – Memory Fine Base Address High Register 1

Address FFFFFFFE10– Memory Fine Base Address High Register 2

Address FFFFFFFE18 – Memory Fine Base Address High Register 3

Address FFFFFFFE20 – Memory Fine Base Address High Register 4

Address FFFFFFFE28 – Memory Fine Base Address High Register 5

Bit Number	15:0
Bit Name	ADDRESS[31:16]
Access	R/W
Default	0000_0000_0000_0000

Bits 15-0: ADDRESS[31:16] – 16 Most Significant Bits of the Base Address. The Base Address sets the 22 most significant bits of the memory address.

5.4 Memory Fine Base Address Low Register 1-5 (MFBALRx)

Address FFFFE0C – Memory Fine Base Address Low Register 1

Address FFFFE14 – Memory Fine Base Address Low Register 2

Address FFFFE1C – Memory Fine Base Address Low Register 3

Address FFFFE24 – Memory Fine Base Address Low Register 4

Address FFFFE2C – Memory Fine Base Address Low Register 5

Bit Number	15:10	9	7:4	1	0
Bit Name	ADDRESS[15:10]	AW	BLOCK_SIZE	RONLY	PRIV
Access	R/W	R/W	R/W	R/W	R/W
Default	000000	0	0000	0	0

Bits 15-10: ADDRESS[15:10] – 6 Least Significant Bits of the Base Address. The Base Address sets the 22 most significant bits of the memory address.

Bit 9: AW – Auto-wait-on-write. When this bit is set, any write operation on this memory select takes two system cycles.

0 = Write operation is not supplemented with an additional cycle (Default)

1 = Write operation takes an additional cycle

Bits 7-4: BLOCK_SIZE – Configures the size of the memory

0000 = Memory select is disabled (Default)

0001 = 1K Bytes

0010 = 2K Bytes

0011 = 4K Bytes

0100 = 8K Bytes

0101 = 16K Bytes

0110 = 32K Bytes

0111 = 64K Bytes

1000 = 128K Bytes

1001 = 256K Bytes

1010 = 512K Bytes

1011 = 1M Bytes

1100 = 2M Bytes

1101 = 4M Bytes

1110 = 8M Bytes

1111 = 16M Bytes

Bit 1: RONLY – Read-only protection. This bit sets read-only protection for the memory selected by the memory select. An illegal access exception is generated when a write is attempted to the memory.

0 = Read/write access to memory (Default)

1 = Read accesses to memory only

Bit 0: PRIV – Privilege mode protection. This bit sets privilege mode protection for the memory Registration selected by the memory select. An illegal access exception is generated on any access to memory protected by privilege mode.

0 = User/privilege mode accesses to memory (Default)

1 = Privilege mode accesses to memory only

5.5 Program Flash Control Register (PFLASHCTRL)

Address FFFFE60

Bit Number	11	10	9	8	7:5	4:0
Bit Name	BUSY	RSVD	PAGE_ERASE	MASS_ERASE	RESERVED	PAGE_SEL
Access	R	-	R/W	R/W	R/W	R/W
Default	-	0	0	0	000	00000

Bit 11: BUSY – Program Flash Busy Indicator

0 = Program Flash available for read/write/erase access

1 = Program Flash unavailable for read/write/erase access

Bit 10: RSVD – Reserved for future use – leave as a zero

Bit 9: PAGE_ERASE – Program Flash Page Erase Enable

0 = No Page Erase initiated on Program Flash (Default)

1 = Page Erase on Program Flash enabled. Page erased is based on PAGE_SEL (Bits 4-0). This bit is cleared upon completion of Page Erase cycle.

Bit 8: MASS_ERASE – Program Flash Mass Erase Enable

0 = No Mass Erase initiated on Program Flash (Default)

1 = Mass Erase of Program Flash enabled. This bit is cleared upon completion of Mass Erase cycle.

Bits 4-0: PAGE_SEL – Selects page to be erased during Page Erase Cycle

5.6 Data Flash Control Register (DFLASHCTRL)

Address FFFFE64

Bit Number	11	10	9	8	7:6	5:0
Bit Name	BUSY	RSVD	PAGE_ERASE	MASS_ERASE	RESERVED	PAGE_SEL
Access	R	-	R/W	R/W	-	R/W
Default	-	0	0	0	-	000000

Bit 11: BUSY – Data Flash Busy Indicator

0 = Data Flash available for read/write/erase access

1 = Data Flash unavailable for read/write/erase access

Bit 10: RSVD – Reserved for future use – leave as a zero

Bit 9: PAGE_ERASE – Data Flash Page Erase Enable

0 = No Page Erase initiated on Data Flash (Default)

1 = Page Erase Cycle on Data Flash enabled. Page erased is based on PAGE_SEL (Bits 4-0). This bit is cleared upon completion of Page Erase cycle.

Bit 8: MASS_ERASE – Data Flash Mass Erase Enable

0 = No Mass Erase initiated on Data Flash (Default)

1 = Mass Erase of Data Flash enabled. Bit is cleared upon completion of mass erase.

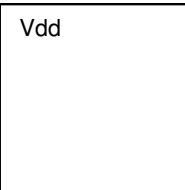
Bits 5-0: PAGE_SEL – Selects page to be erased during Page Erase Cycle

5.7 Program Flash Interlock Register (PFLASHILOCK)

Address FFFFE68

Bit Number	31:0
Bit Name	INTERLOCK_KEY
Access	R/W
Default	0000_0000_0000_0000_0000_0000_0000_0000

Bit 31-0: INTERLOCK_KEY – Program Flash Interlock Key. Register must be set to 0x42DC157E prior to every Program Flash write. If the Interlock Key is not set, the write to the



8/18/2010
Product Preview

Address Manager (DEC) Programmer's Guide
UCD30xx

Version 0.1

Program Flash will not proceed. This register will clear upon the completion of a write to the Program Flash.