



APPLICATION NOTE 3563

Using Data Pointers to Read/Write to SRAM

Abstract: The MAXQ-based microcontroller uses data pointers to read and write to SRAM. This application note describes how to move data from program memory to the SRAM, and how to access the data from SRAM using the data pointers.

Introduction

This application note describes how to write two words of data to SRAM, using data pointers DP[0] and BP [OFFS]. The data is then read back into the MAXQ's accumulators from SRAM using the same data pointers, and a logical "AND" is performed on the two registers.

Getting Started

To begin, you need basic knowledge about the MAXQ architecture, register map, and instruction set, which can be obtained from the MAXQ Family User's Guide or from any MAXQ-based microcontroller data sheet. A good example is the MAXQ2000 data sheet. You also need to reference "Accessing Data Memory" which is in the MAXQ Family User's Guide. Basic familiarity with assembly language in general, and with MAXQ assembler in particular, is assumed.

Data Pointer Overview

There are three data pointers available on a MAXQ-based microcontroller: DP[0], DP[1], and BP[OFFS]. Each of these data pointers can be configured to either word- or byte-access modes by setting the corresponding bit in the Data Point Control (DPC) register.

Name	Function
DP[0]	Data pointer 0
DP[1]	Data pointer 1
BP[OFFS]	Frame pointer base
OFFS	Offset of frame pointer base
DPC	Data pointer control
SDPS0 SDPS1	00b selects DP[0] as active pointer 01b selects DP[1] as active pointer 10b selects BP as active pointer
WBS0	DP[0] word mode WBS0=1, byte mode WBS0=0
WBS1	DP[1] word mode WBS1=1, byte mode WBS0=0
WBS2	BP word mode WBS2=1, byte mode WBS0=0

Register	Bit Position
DP[0]	DP[0] (16 bits)
DP[1]	DP[1] (16 bits)
BP	BP = BP[Offs] (16 bits)
OFFS	- - - - OFFS (8 Bits)
DPC	DPC (16 bits)
	- - - WBS2 WBS1 WBS0 SDPS1 SDPS0

The three pointers share a single read/write port on the data memory, and thus the user must knowingly activate the desired pointer before accessing memory. This can be done explicitly using the data select bits (SDPS2:0; DPC.1:0), or implicitly by writing to the DP[n], BP, or OFFS register as shown below. Any indirect memory access using a data pointer will also set the SDPS bits, thus activating the pointer as the active source pointer.

```

move DPC,    #2           ;(explicit) selection of BP as the active pointer
move DP[1], DP[1]       ;(implicit) selection of DP[1]; set SDPS1: 0=01b
move OFFS,   src        ;(implicit) selection of BP; set SDPS1=1
move @DP[0],src        ;(implicit) selection of DP[0]; set SDPS1:0=00b

```

Increment/Decrement Data Pointer

Data pointers can be updated using pre- and post-increment/decrement operator with a register or a virtual NUL destination. Data pointer increment/decrement operation can be done as follows:

```

move NUL, @DP[0]++      ;increment DP[0]
move NUL, @DP[1]--      ;decrement DP[1]
move NUL, @BP[OFFS++]   ;increment Frame Pointer Base + Offset + 1

move @++DP[0], A[1]     ;increment address and store A[1] at new address
;Note: Only the pre-increment/decrement can be ;used
;when writing to memory

move A[1], @DP[1]--     ;Reads value from DP[1] and store in A[1] then
;decrement

;Note: Only the post-increment/decrement can be
;used

;when reading from memory

```

Code Example

1. Move 2 words to SRAM using DP[0]. 5555h is moved to the first word of SRAM, referenced by int_Var1; AAAAh is moved to the second word of SRAM, referenced by int_Var2. Var_1 and Var_2 are read back from SRAM into accumulator A[0] and A[1].

```

int_Var1 EQU 0h          ;address of int_Var1 to the first byte of SRAM
int_Var2 EQU 1h          ;address of int_Var2 to the second byte of SRAM

main:
move DPC,    #4h         ;Set DP[0] to word mode
;Must be set active before using

move DP[0],  #int_Var1   ;Load address of int_Var1 into DP[0]
;also activated DP[0]

```

```

move @DP[0], #5555h           ;write 5555h to SRAM at address of int_Var1

move DP[0], #int_Var2 ;Load address into of int_Var1 into DP[0]
move @DP[0], #0AAAAh     ;Writes AAAAh to SRAM at address of int_Var2

move DP[0], #int_Var1 ;Load address of int_Var1 into DP[0]
move A[0], @DP[0]     ;Reads from address of int_Var1

move DP[0], #int_Var2 ;Load address of int_Var2 into DP[0]
move A[1], @DP[0]     ;Reads from address of int_Var2

move AP, #0             ;select accumulator 0
and A[1]                ;AND the A[1] and A[0] and store in A[0]
end

```

2. Move 2 bytes to SRAM using DP[1]. 55h is moved to the first byte of SRAM, referenced by int_Var1; AAh is moved to the second byte of SRAM, referenced by int_Var2. Var_1 and Var_2 are read back from SRAM into accumulator A[0] and A[1].

```

int_Var1 EQU 0h           ;address of int_Var1 to the first word of SRAM
int_Var2 EQU 1h           ;address of int_Var2 to the second word of SRAM

main:
move DPC, #0h             ;Set DP[1] to byte mode
                           ;Must be set active before using

move DP[1], #int_Var1 ;Load address of int_Var1 into DP[1]
                           ;also activated DP[1]
move @DP[1], #55h       ;write 55h to SRAM at address of int_Var1

move DP[1], #int_Var2 ;Load address into of int_Var1 into DP[1]
move @DP[1], #0AAh     ;Writes AAh to SRAM at address of int_Var2

move DP[1], #int_Var1 ;Load address of int_Var1 into DP[1]
move A[0], @DP[1]     ;Reads from address of int_Var1

move DP[1], #int_Var2 ;Load address of int_Var2 into DP[1]
move A[1], @DP[1]     ;Reads from address of int_Var2

move AP, #0             ;select accumulator 0
and A[1]                ;AND the A[1] and A[0] and store in A[0]
end

```

3. Move 2 words to SRAM using BP[Offs]. 5555h is moved to the first word of SRAM, referenced by int_Var1; AAAAh is moved to the second word of SRAM, referenced by int_Var2. Var_1 and Var_2 are read back from SRAM into accumulator A[0] and A[1].

```

int_Var1 EQU 0h           ;address of int_Var1 to the first word of SRAM
int_Var2 EQU 1h           ;address of int_Var2 to the second word of SRAM

main:
move DPC, #10h           ;Set BP[OFFS] to word mode
move BP, #0h             ;Sets BP to 0 and activates BP[OFFS]

move OFFS, #int_Var1     ;Load address of int_Var1 into OFFS
move @BP[OFFS], #5555h ;write to 5555h to SRAM at address of int_Var1

```

```

move OFFS, #int_Var2           ;Load address into of int_Var1 into OFFS
move @BP[OFFS], #0AAAAh       ;Writes AAAAh to SRAM at address of int_Var2

move OFFS, #int_Var1           ;Load address of int_Var1 into OFFS
move A[0], @BP[OFFS]          ;Reads from address of int_Var1

move OFFS, #int_Var2           ;Load address of int_Var2 into OFFS
move A[1], @BP[OFFS]          ;Reads from address of int_Var2

move AP, #0                    ;select accumulator 0
and A[1]                        ;AND the A[1] and A[0] and store in A[0]
end

```

Invalid Data Pointer Instruction

The following are incorrect uses of data pointers:

Assembly Instruction	Problem	Solution
<pre> move DP[0], DP[0] move A[1], @DP[0] move A[2], @DP[1] </pre>	Data pointer initialized to DP[0]. Needs to be reinitialized to DP[1].	<pre> move DP[0], DP[0] move A[1], @DP[0] move DP[1], DP[1] move A[2], @DP[1] </pre>
<pre> move BP[OFFS], #0h </pre>	BP and OFFS must be initialized separately.	<pre> move BP, #0h move OFFS, #0h </pre>
<pre> move @DP[0]++, A[0] </pre>	Only the pre-increment/decrement can be used.	<pre> move @++DP[0], A[0] </pre>
<pre> move A[0], @++DP[0] </pre>	Only the post-increment/decrement can be used.	<pre> move A[0], @DP[0]-- </pre>
<pre> move @++DP[0], @DP[0]++ </pre>	Unable to increment data pointer and store it using the same data pointer.	<pre> move A[1], @DP[0]++ move @++DP[0], A[1] </pre>
<pre> move @DP[0]++, A[1] </pre>	You cannot use post-increment/decrement for a destination.	<pre> move @DP[0], A[1] move NUL, @DP[0]++ </pre>
<pre> move A[0], @--DP[0] </pre>	You cannot use pre-increment/decrement for a source.	<pre> move NUL, @DP[0]-- move A[0], @DP[0] </pre>

Relevant Links

- [MAXQ Home Page](#)
- [MAXQ Family User's Guide](#)
- [MAXQ2000 User's Guide Supplement](#)
- [Dallas Semiconductor Microcontroller Support Forum](#)

Application Note 3563: www.maxim-ic.com/an3563

More Information

For technical support: www.maxim-ic.com/support

For samples: www.maxim-ic.com/samples

Other questions and comments: www.maxim-ic.com/contact

Automatic Updates

Would you like to be automatically notified when new application notes are published in your areas of interest?

[Sign up for EE-Mail™.](#)

Related Parts

MAXQ2000: [QuickView](#) -- [Full \(PDF\) Data Sheet](#) -- [Free Samples](#)

AN3563, AN 3563, APP3563, Appnote3563, Appnote 3563

Copyright © by Maxim Integrated Products

Additional legal notices: www.maxim-ic.com/legal