Introducing the Enhanced Mid-Range Architecture

© 2006 Microchip Technology Incorporated. All Rights Reserved.

Hello, my name is Marc McComb Technical Training Engineer in the Security Microcontroller Technology Division at Microchip. Thank you for viewing this web seminar introducing Microchip's newly enhanced Mid-Range 8-bit PIC® Microcontrollers.

MICROCHIP
WebSeminars

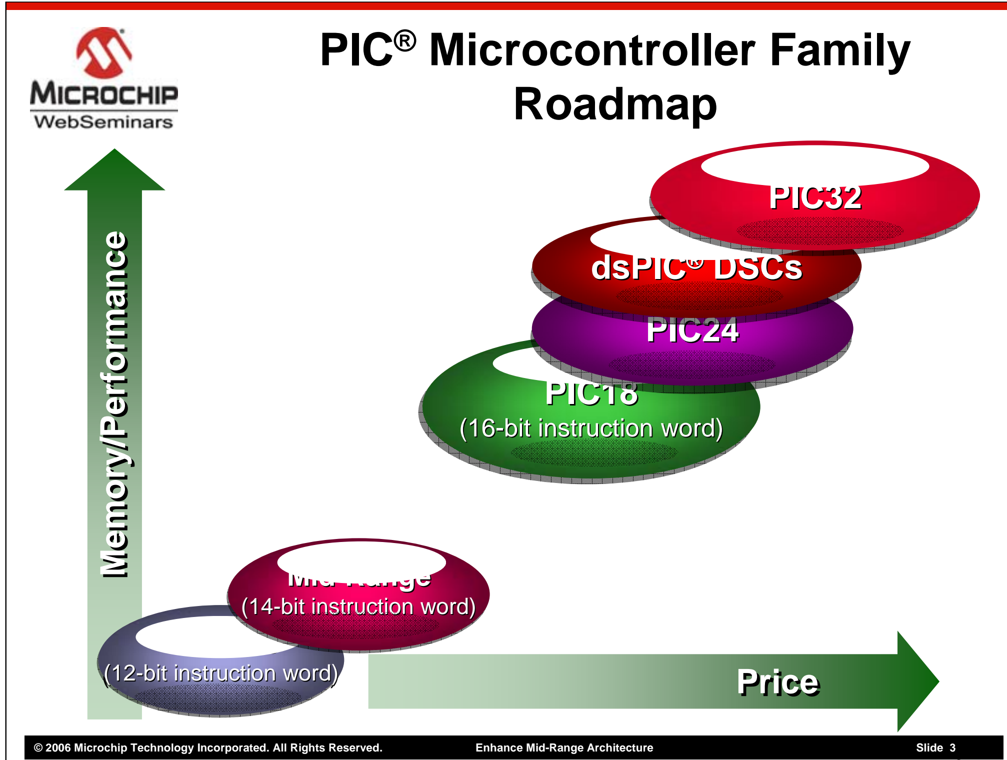
Agenda

- Introducing the Enhanced Mid-Range Architecture
- Enhancements to the Data Memory Map
- New Instructions
- Indirect Addressing Enhancements
- Migration Considerations

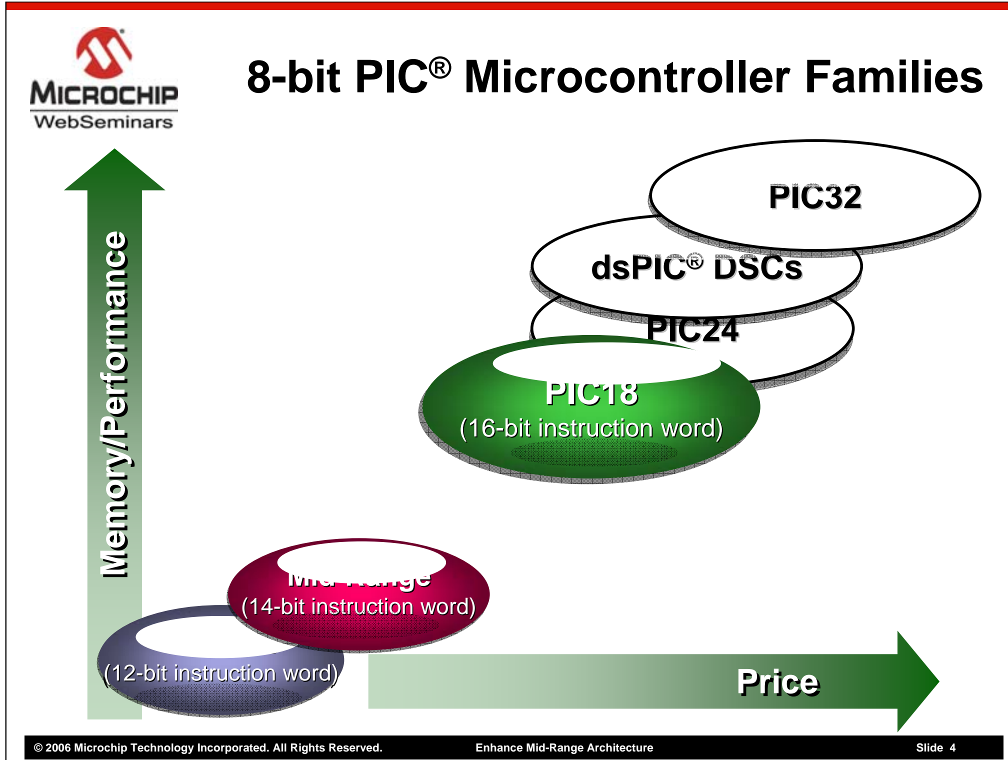
© 2006 Microchip Technology Incorporated. All Rights Reserved.

We will cover a number of topics during the course of this web seminar. First, an overview of the enhancements made to the 8-bit mid-range PIC architecture. Next, a detailed description of some of the enhancements to the Data Memory Map, indirect memory addressing and some of the new instructions added. Finally, I will describe what should be considered when migrating designs from the current architecture to the new enhanced core.

So let’s begin…



Microchip currently offers a wide variety of 8-bit to 32-bit microcontrollers along with digital signal controllers. These products range from the very small, low-cost 8-bit Baseline family all the way up to the sophisticated dsPIC® DSCs with powerful high-level digital signal processing capabilities.

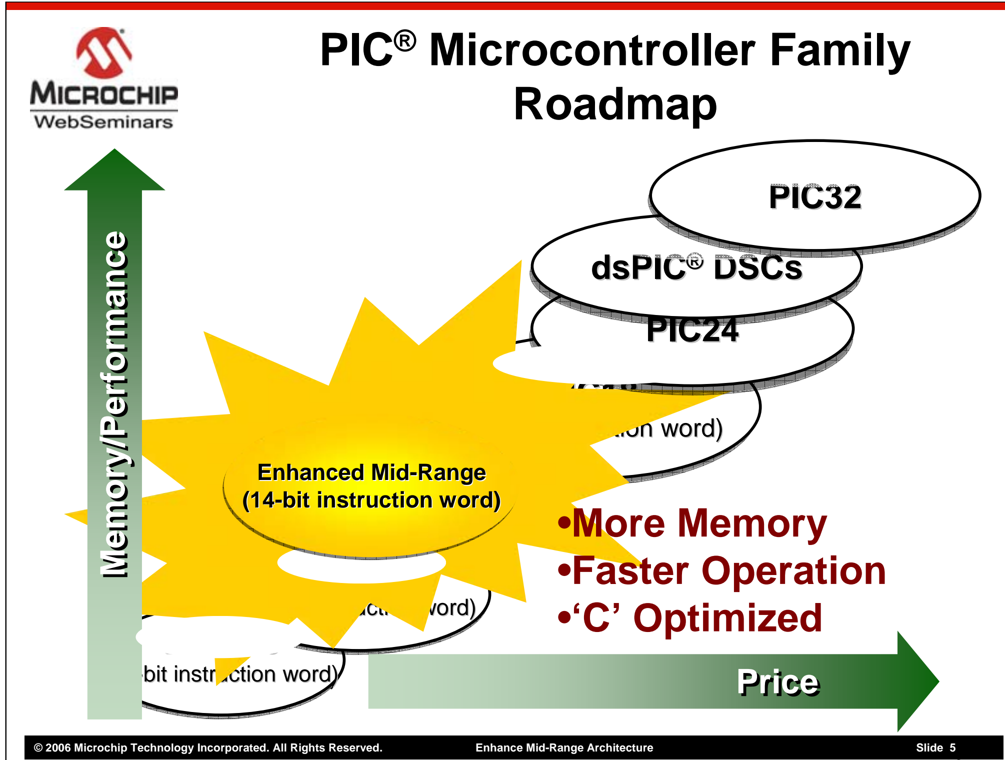


Our 8-bit MCUs, including the PIC10, PIC12, PIC16 and PIC18 families, offer the designer a range of choices with variations in performance, memory and pin count. Baseline Products offer a 12-bit instruction set covering 6-40-Pin and up to 3 kByte program memory or 2k instructions with a basic peripheral set including comparators and Analog-to-Digital Converters.

Our existing Mid-Range family offers a 14-bit instruction set covering 8-64-Pin with up to 14 kByte program memory or 8k instructions. With up to 5 MIPS performance, a more advanced peripheral set is offered including Serial Communications, Capture/Compare/PWM, LCD, EEPROM, 10-bit A/D and, more recently, our Capacitive mTouch™ Sensing peripherals.

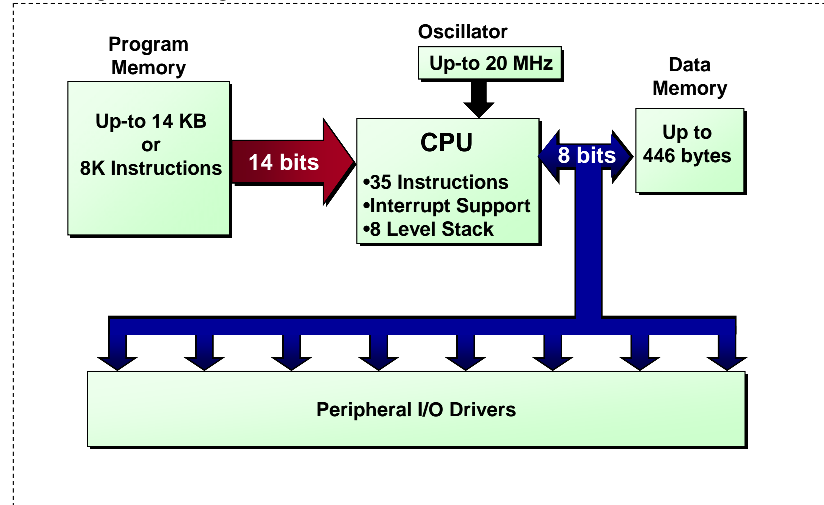
Finally, the high-performance PIC18 products offer a 16-bit instruction set covering 18-100-Pin with up to 128 kByte program memory and up to 16 MIPS performance with hardware multiply. This family offers a very advanced peripheral set including Advanced Serial Communications (CAN, USB), Capture/Compare/PWM, LCD, USB, Ethernet, 12-bit A/D and the new Charge Time Measurement Peripheral developed specifically for Capacitive Sensing Applications.

Recognizing the demand for increased performance and peripherals within the 8-bit MCU market, Microchip has recently revisited and enhanced the Mid-Range Architecture.



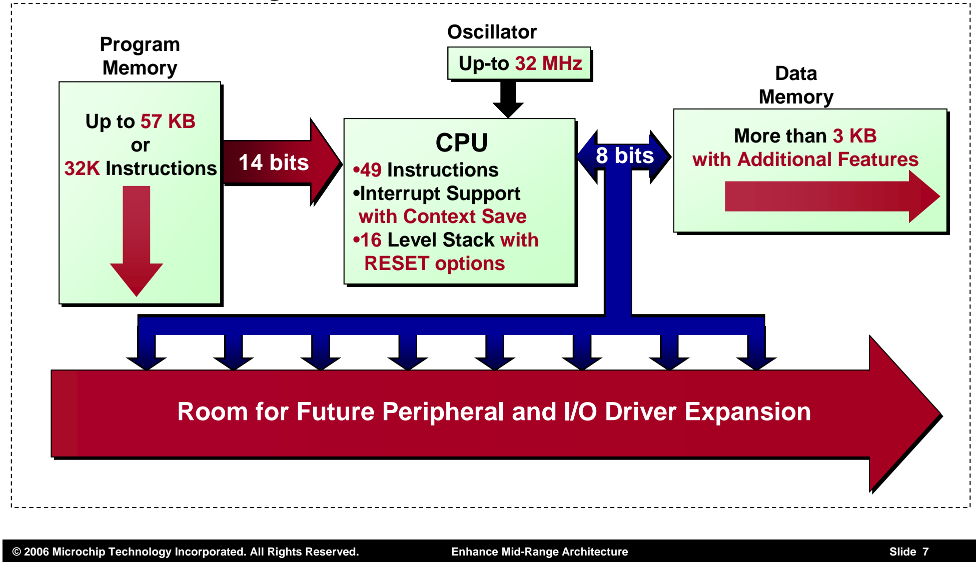
The new enhanced core builds upon the best elements of the existing Mid-Range core and provides additional performance, while maintaining compatibility with existing Mid-Range products for true product migration. The enhancements provide users with a boost of performance of up to 50% and code-size reductions of up to 40% for various algorithms and functions. Improvements such as more program and data memory including "C" efficiency optimizations, increased peripheral support, and reduced interrupt latency.

Customers now have more options for applications that may not require the extended 16-bit program memory sizes offered by the PIC18 family but could benefit from more capabilities than currently offered by the existing Mid-Range architecture.



Shown here is a simplified block diagram of an 8-bit Mid-Range PIC® microcontroller. This architecture incorporates two separate memory spaces for up to 8K of program instructions and 446 bytes of data. Separate busses, one for each memory space, can be active simultaneously; therefore, while fetching an instruction on the program bus, data can be read or written on the data bus. The Program Memory bus is optimized for a 14-bit instruction word so that many instructions can be executed in a single instruction cycle.

The Central Processing Unit on the existing architecture, supports 35 instructions with interrupt capabilities using an 8 level hardware stack.



The enhanced Mid-Range architecture takes a good thing and makes it better. Memory space is significantly expanded, operating speed is improved up to 8 MIPS, a 60% increase over the existing architecture, and new instructions reduce overhead and optimize instruction execution times. Additionally, there is a lot more room for peripheral expansion as new technologies emerge. Let's take a look at a side-by-side comparison with the existing architecture and then go into more detail on these enhancements.

Side-by-Side Comparison

	Existing	Enhanced
Instruction Length	14 (bits)	14 (bits)
Addressable Program Memory	8K (Instructions)	32K (Instructions)
Max RAM and Register space	446 (Bytes)	> 3K (Bytes)
Instruction Count	35	49
Hardware Stack	8	16 With optional RESET Capabilities
Interrupt Handling	Software	Hardware
File Select Registers	1x9 bit	2x16 bit

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 8

While maintaining the instruction word at 14 bits, addressable program memory has been extended to 57 Kbytes, allowing up to 32K of instructions on the enhanced Mid-Range architecture. This is a dramatic increase from the existing architecture at 8K instructions. The user now has a lot more room for more involved firmware applications along with optimization when programming in C.

Data Memory has been extended to more than 3 Kbytes from 446 bytes on the existing architecture. As I mentioned, not only does this increase available General Purpose RAM, but now there is more room for future peripheral expansion.

With the new enhanced architecture comes 14 new instructions that enable the embedded engineer to optimize firmware applications by implementing faster code with less overhead.

The hardware stack has been increased from 8 to 16 levels, with optional reset capabilities on overflow or underflow conditions. Furthermore, the stack is now completely accessible from Bank 31 in the extended Data Memory space. More on this later.

Another nice feature is the new hardware context save during interrupts. Traditionally, a software handling routine is required on the existing architecture to save some of the core registers during an interrupt. Now this is done automatically with these registers saved on any interrupt and restored on a return.

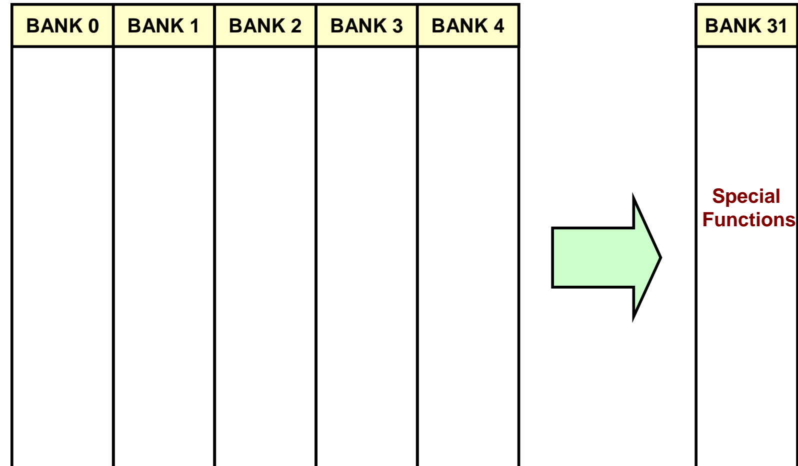
Finally, indirect addressing has been enhanced with the addition of two 16-bit file select registers allowing not only access to data memory but program memory as well. Additionally, three new instructions are included in the new instruction set specifically for FSR support.

MICROCHIP
WebSeminars

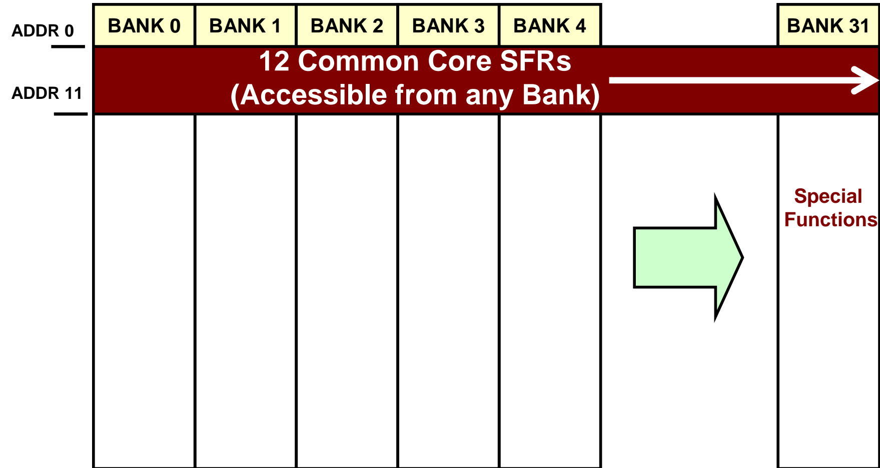
Data Memory

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhanced Mid-Range Architecture Slide 9

Let's take a closer look at these enhancements starting with the new memory map.



Users familiar with the Mid-Range PIC® Microcontroller architecture may remember that the PIC Microcontroller Data Memory is organized into 4 banks of registers. On the new enhanced architecture, data memory is now expanded to 32 banks with BANK 31 offering some special functions.



The first 12 bytes of every bank are now identical and will hold the core or CPU specific registers. To access any register in the data memory map, when programming in assembly, the user needs to ensure that the bank holding the target register is selected in firmware before it can be accessed, this is called banking. With the core registers common across all banks, they can be easily accessed from any bank minimizing the need for banking thereby reducing code overhead. So what are these registers?

MICROCHIP
WebSeminars

Common Core Registers

	Address	Register	Function
	0x00	INDF0	Indirect Register 0
NEW	0x01	INDF1	Indirect Register 1
	0x02	PCL	Program Counter Low
	0x03	STATUS	Status Register
NEW	0x04	FSR0 Low	File Select Register 0 Low Byte
NEW	0x05	FSR0 High	File Select Register 0 High Byte
NEW	0x06	FSR1 Low	File Select Register 1 Low Byte
NEW	0x07	FSR1 High	File Select Register 1 High Byte
NEW	0x08	BSR	Bank Select Register
NEW	0x09	WREG	Working Register
	0x0A	PCLATH	Program Counter Latch High
	0x0B	INTCON	Interrupt Control Register

© 2006 Microchip Technology Incorporated. All Rights Reserved.

Here are the common core registers on the new enhanced architecture. I have highlighted some new additions. Let’s take a look.

MICROCHIP
WebSeminars

Common Core Registers

	Address	Register	Function
	0x00	INDF0	Indirect Register 0
NEW	0x01	INDF1	Indirect Register 1
	0x02	PCL	Program Counter Low
	0x03	STATUS	Status Register
NEW	0x04	FSR0 Low	File Select Register 0 Low Byte
NEW	0x05	FSR0 High	File Select Register 0 High Byte
NEW	0x06	FSR1 Low	File Select Register 1 Low Byte
NEW	0x07	FSR1 High	File Select Register 1 High Byte
NEW	0x08	BSR	Bank Select Register
NEW	0x09	WREG	Working Register
	0x0A	PCLATH	Program Counter Latch High
	0x0B	INTCON	Interrupt Control Register

© 2006 Microchip Technology Incorporated. All Rights Reserved.

There are now two 16-bit File Select Registers, or FSRs, used for computing memory addresses at run time allowing data modification using what is called indirect addressing. Since data memory is an 8-bit register space, each 16-bit FSR will consist of two 8-bit registers: FSR Low and FSR High. Along with the new FSR registers are the associated 8-bit indirect file registers INDF0 and INDF1.

MICROCHIP WebSeminars

Common Core Registers

	Address	Register	Function
	0x00	INDF0	Indirect Register 0
NEW	0x01	INDF1	Indirect Register 1
	0x02	PCL	Program Counter Low
	0x03	STATUS	Status Register
NEW	0x04	FSR0 Low	File Select Register 0 Low Byte
NEW	0x05	FSR0 High	File Select Register 0 High Byte
NEW	0x06	FSR1 Low	File Select Register 1 Low Byte
NEW	0x07	FSR1 High	File Select Register 1 High Byte
NEW	0x08	BSR	Bank Select Register
NEW	0x09	WREG	Working Register
	0x0A	PCLATH	Program Counter Latch High
	0x0B	INTCON	Interrupt Control Register

5-bit register → Writing to this register selects a Bank (0 → 31) in a single cycle

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 14

A new bank select register has been added. Writing to this 5-bit register selects any of the 32 memory banks in a single instruction clock cycle a reduction of one clock cycle from the traditional method using the existing architecture. This register also comes with a new associated instruction that will be discussed later.

MICROCHIP WebSeminars

Common Core Registers

	Address	Register	Function
	0x00	INDF0	Indirect Register 0
NEW	0x01	INDF1	Indirect Register 1
	0x02	PCL	Program Counter Low
	0x03	STATUS	Status Register
NEW	0x04	FSR0 Low	File Select Register 0 Low Byte
NEW	0x05	FSR0 High	File Select Register 0 High Byte
NEW	0x06	FSR1 Low	File Select Register 1 Low Byte
NEW	0x07	FSR1 High	File Select Register 1 High Byte
NEW	0x08	BSR	Bank Select Register
NEW	0x09	WREG	Working Register
	0x0A	PCLATH	Program Counter Latch High
	0x0B	INTCON	Interrupt Control Register

© 2006 Microchip Technology Incorporated. All Rights Reserved.

You may have noticed that the working register is now included in the data memory map. The new enhanced architecture now makes the working register completely addressable.

MICROCHIP WebSeminars

Common Core Registers

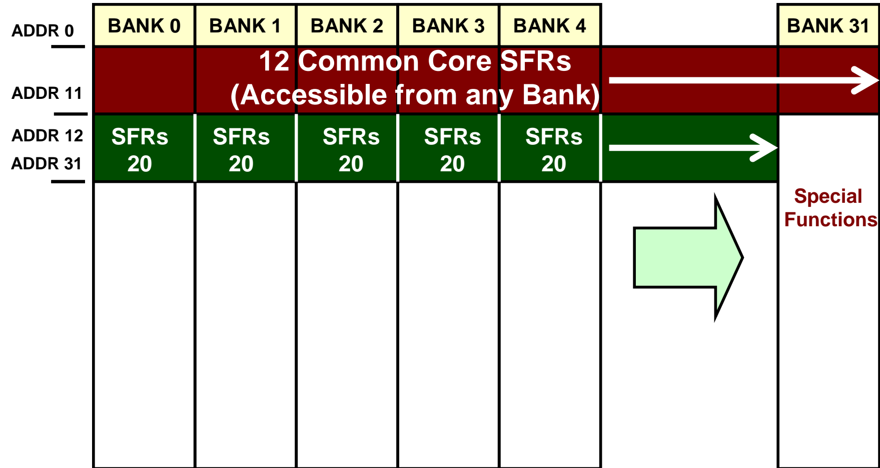
	Address	Register	Function
	0x00	INDF0	Indirect Register 0
NEW	0x01	INDF1	Indirect Register 1
	0x02	PCL	Program Counter Low
	0x03	STATUS	Status Register
NEW	0x04	FSR0 Low	File Select Register 0 Low Byte
NEW	0x05	FSR0 High	File Select Register 0 High Byte
NEW	0x06	FSR1 Low	File Select Register 1 Low Byte
NEW	0x07	FSR1 High	File Select Register 1 High Byte
NEW	0x08	BSR	Bank Select Register
NEW	0x09	WREG	Working Register
	0x0A	PCLATH	Program Counter Latch High
	0x0B	INTCON	Interrupt Control Register

Automatically Saved on Interrupt
Automatically Restored on RETFIE

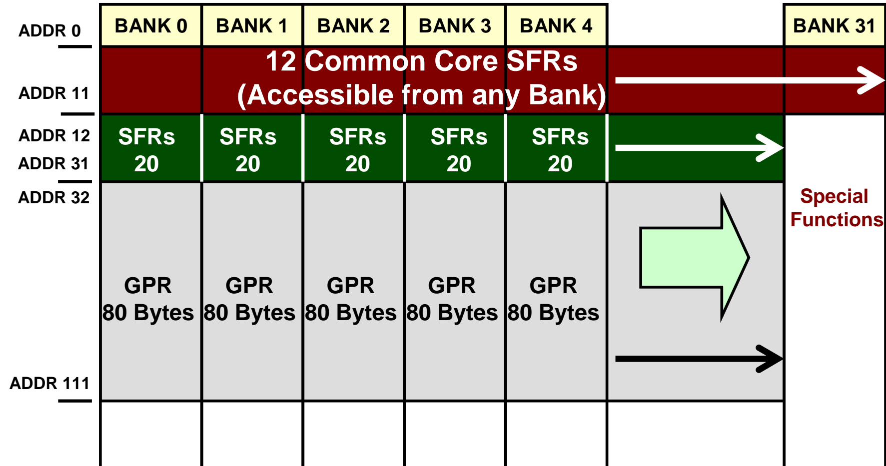
© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 16

Traditionally, on the existing architecture, to save the context of such registers as the working, status and PCL register, the user simply copied the code from the Microchip supplied template into the firmware. On any interrupt, this software routine saved the required core registers then performed the interrupt service routine.

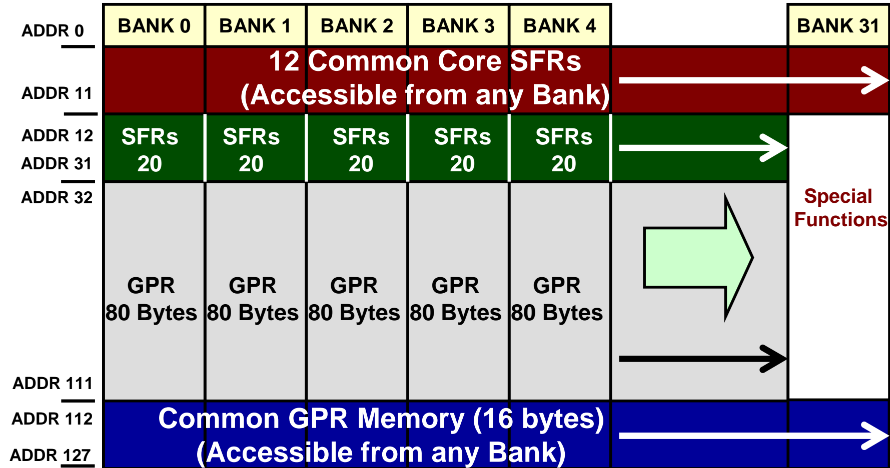
The Enhanced Mid-Range Architecture now does this context save in hardware eliminating the need of the software overhead. On any interrupt, the contents of these registers are automatically saved. The RETFIE or Return from Interrupt Enable instruction automatically restores the context in addition to setting the global interrupt enable bit GIE.



As I mentioned earlier, with an extended memory map, there is now a lot more room for peripheral specific Special Function Registers or SFRs. In the enhanced architecture, 20 SFRs immediately follow the 12 common core registers in each bank up to bank 30.



The next 80 bytes of banks 0 to 30 hold the general purpose ram. These registers hold such data as variables created and used by the firmware. This expansion further enhances the data memory space for such high level languages as C that require room in data memory for variables created in the background. When we look at indirect addressing enhancements, we'll see that these registers shadowed or mirrored as a linear General Purpose Ram region in the File Select Register Memory Map.



Another nice addition, the last 16 address spaces in General Purpose Ram is common across all banks. Similar to the 12 common core Special Function Registers, any data written from address 112 to 127 is accessible from all 32 banks further minimizing the need for banking when targeting these registers.

Bank 31 Special Functions

MICROCHIP WebSeminars

- Provides Access to:
  - Stack Access and Debugging Registers
  - Access to context registers saved during interrupts

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 20

Bank 31 of the enhanced data memory space offers some features for advanced firmware applications. First, the user now has full access to the 16 levels of the Hardware Stack for read/write operations via some new stack registers. This enhancement allows such neat tricks as changing the return location during an interrupt or creating sub-stacks for different operations.

Another feature of bank 31 relates to the automatic hardware context save. A section of bank 31 registers contain a backup copy of the saved context and are the source used when the context is restored on a return from interrupt. Therefore, if these registers need to be adjusted during an interrupt they can be accessed here.

As I mentioned, these are advanced functions that are only briefly highlighted here. For more information, the data sheet for the specific microcontroller used and the resources listed at the end of this presentation should all be referenced.

MICROCHIP
WebSeminars

New Instructions

© 2006 Microchip Technology Incorporated. All Rights Reserved.

That's a brief overview of the enhanced Data Memory Map. Let's now turn our attention to the new instructions added and associated benefits to the user.

MICROCHIP WebSeminars

New Instructions

Mnemonic	Description
ADDWFC	Add W+F with Carry
SUBWFB	Subtract F-W with Borrow
LSLF	Logical Shift Left
LSRF	Logical Shift Right
ASRF	Arithmetic Shift Right
MOVLP	Move Literal to PCLATH
MOVLB	Move Literal to BSR
BRA	Branch Relative (signed)
BRW	Branch PC + W (unsigned)
CALLW	Call PCLATH:W
ADDFSR	Add Literal to FSRn (signed)
MOVIW	Move indirect to W
MOVWI	Move W to Indirect
RESET	Reset Hardware & Software

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 22

The enhanced Mid-Range architecture adds 14 new instructions to the existing architecture's 35 instructions bringing the total instruction count to 49. These additional instructions improve arithmetic, simplify paging and banking, and extend the capabilities of the indirect addressing modes.

The new instructions shown here can be categorized into functional types.

MICROCHIP
WebSeminars

New Instructions

- **New Arithmetic and Logic Shifts Instructions**

Mnemonic	Description
ADDWFC	Add W+F with Carry
SUBWFB	Subtract F-W with Borrow
LSLF	Logical Shift Left
LSRF	Logical Shift Right
ASRF	Arithmetic Shift Right
MOVLP	Move Literal to PCLATH
MOVLB	Move Literal to BSR
BRA	Branch Relative (signed)
BRW	Branch PC + W (unsigned)
CALLW	Call PCLATH:W
ADDFSR	Add Literal to FSRn (signed)
MOVIW	Move indirect to W
MOVWI	Move W to Indirect
RESET	Reset Hardware & Software

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 23

Arithmetic operations on the Mid-Range architecture has always been easy, but multi-byte math such as 16 or 32-bit operations are now made easier with new instructions that support carry/borrow, add and subtract along with arithmetic shifts.

The logical shift left and right instructions perform shifts similar to the legacy rotate instructions with the exception that the incoming digit is not drawn from the carry flag but is a 0.

The arithmetic shift right instruction performs a sign extension while shifting the remaining 7 bits. Therefore, a positive value remains positive and negative value remains negative. Notice that there is no arithmetic shift left. As you may have guessed, this would be the same as a logical shift left.

MICROCHIP
WebSeminars

New Instructions

- New Arithmetic and Logic Shifts Instructions
- **Single-Cycle Paging and Banking**

Mnemonic	Description
ADDWFC	Add W+F with Carry
SUBWFB	Subtract F-W with Borrow
LSLF	Logical Shift Left
LSRF	Logical Shift Right
ASRF	Arithmetic Shift Right
MOVLP	Move Literal to PCLATH
MOVLB	Move Literal to BSR
BRA	Branch Relative (signed)
BRW	Branch PC + W (unsigned)
CALLW	Call PCLATH:W
ADDFSR	Add Literal to FSRn (signed)
MOVIW	Move indirect to W
MOVWI	Move W to Indirect
RESET	Reset Hardware & Software

© 2006 Microchip Technology Incorporated. All Rights Reserved.

Those users familiar with the overhead created when paging or banking on the existing architecture will surely welcome the addition of these next two instructions. The Move Literal to PCLATH instruction, when executed, moves the program counter to a new page in a single instruction cycle reducing the paging overhead while not disturbing the working register.

Remember the Bank Select Register discussed in the new core register section of this presentation? The move literal to bank select register, when executed, loads the bank select register which in turn changes to the appropriate bank. All this is done in one instruction cycle while leaving the working register alone.

MICROCHIP WebSeminars

New Instructions

- New Arithmetic and Logic Shifts Instructions
- Single-Cycle Paging and Banking
- **Relative Branching and Faster Table Reads**

Mnemonic	Description
ADDWFC	Add W+F with Carry
SUBWFB	Subtract F-W with Borrow
LSLF	Logical Shift Left
LSRF	Logical Shift Right
ASRF	Arithmetic Shift Right
MOVLP	Move Literal to PCLATH
MOVLB	Move Literal to BSR
BRA	**Branch Relative (signed)**
BRW	**Branch PC + W (unsigned)**
CALLW	**Call PCLATH:W**
ADDFSR	Add Literal to FSRn (signed)
MOVIW	Move indirect to W
MOVWI	Move W to Indirect
RESET	Reset Hardware & Software

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 25

Relative branching is branching to a target address that is based upon the current address. The enhanced Mid-Range architecture adds a branch relative and branch program counter using the working register value as the offset. The branch relative instruction is signed and therefore allows branching either forward or backwards from the current program counter value by 256 instructions in the Program Memory Space. The branch program counter with working register value offset is also a relative branch. However, since it is not signed it will branch from an offset of 0 to 255. This allows traditional table reads to execute much faster without regard for table location.

The callw instruction has been added to further optimize the architecture for C programming languages providing support for function pointers.

Examples of each instruction listed here along with comparisons to previous solutions on the existing core are described in detail within the resources listed at the end of this presentation.

MICROCHIP
WebSeminars

New Instructions

- New Arithmetic and Logic Shifts Instructions
- Single-Cycle Paging and Banking
- Relative Branching and Faster Table Reads
- **FSR Instruction Support**

Mnemonic	Description
ADDWFC	Add W+F with Carry
SUBWFB	Subtract F-W with Borrow
LSLF	Logical Shift Left
LSRF	Logical Shift Right
ASRF	Arithmetic Shift Right
MOVLP	Move Literal to PCLATH
MOVLB	Move Literal to BSR
BRA	Branch Relative (signed)
BRW	Branch PC + W (unsigned)
CALLW	Call PCLATH:W
ADDFSR	Add Literal to FSRn (signed)
MOVIW	Move indirect to W
MOVWI	Move W to Indirect
RESET	Reset Hardware & Software

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 26

The enhanced Mid-Range Architecture now features three new FSR support instructions designed to improve the efficiency of indirect operations. Indirect addressing allows access to arrays and other memory buffers by computing addresses to be computed at run time.

The Add Literal to File Select Register adds a signed value between -32 and +31 to the File Select Register which points to an effective address in memory in a single 16-bit addition.

Using the move indirect to working instruction simply takes the value at the address pointed to by the associated file select register and moves it into the working register. The move working to indirect instructions is the reverse of this by moving the value currently in the working register via the indirect file register into the memory location pointed to by the associated file select register.

MICROCHIP WebSeminars

New Instructions

- New Arithmetic and Logic Shifts Instructions
- Single-Cycle Paging and Banking
- Relative Branching and Faster Table Reads
- FSR Instruction Support
- **New Software RESET**

Mnemonic	Description
ADDWFC	Add W+F with Carry
SUBWFB	Subtract F-W with Borrow
LSLF	Logical Shift Left
LSRF	Logical Shift Right
ASRF	Arithmetic Shift Right
MOVLP	Move Literal to PCLATH
MOVLB	Move Literal to BSR
BRA	Branch Relative (signed)
BRW	Branch PC + W (unsigned)
CALLW	Call PCLATH:W
ADDFSR	Add Literal to FSRn (signed)
MOVIW	Move indirect to W
MOVWI	Move W to Indirect
RESET	**Reset Hardware & Software**

© 2006 Microchip Technology Incorporated. All Rights Reserved.

The RESET instruction performs a complete system RESET. This replaces the previous practice of waiting for the watchdog timer to provide a reset.

MICROCHIP
WebSeminars

Enhanced Indirect Addressing

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhanced Mid-Range Architecture Slide 28

As I have been eluding to throughout this presentation, there are a number of enhancements to indirect addressing. Let's expand on this…

MICROCHIP WebSeminars

(2) 16-bit File Select Registers (FSRs)

Address	Register	Function
0x00	INDF0	Indirect Register 0
0x01	INDF1	Indirect Register 1
0x02	PCL	Program Counter Low
0x03	STATUS	Status Register
0x04	FSR0 Low	File Select Register 0 Low Byte
0x05	FSR0 High	File Select Register 0 High Byte
0x06	FSR1 Low	File Select Register 1 Low Byte
0x07	FSR1 High	File Select Register 1 High Byte
0x08	BSR	Bank Select Register
0x09	WREG	Working Register
0x0A	PCLATH	Program Counter Latch High
0x0B	INTCON	Interrupt Control Register

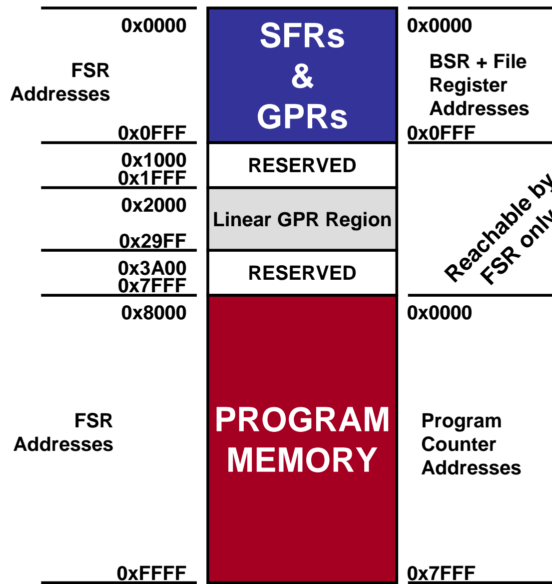
© 2006 Microchip Technology Incorporated. All Rights Reserved.

Referring again to the core registers in the data memory space, the addition of two 16 bit file select registers provides the ability to access additional memory besides the data memory space.

MICROCHIP WebSeminars

FSR Memory Map

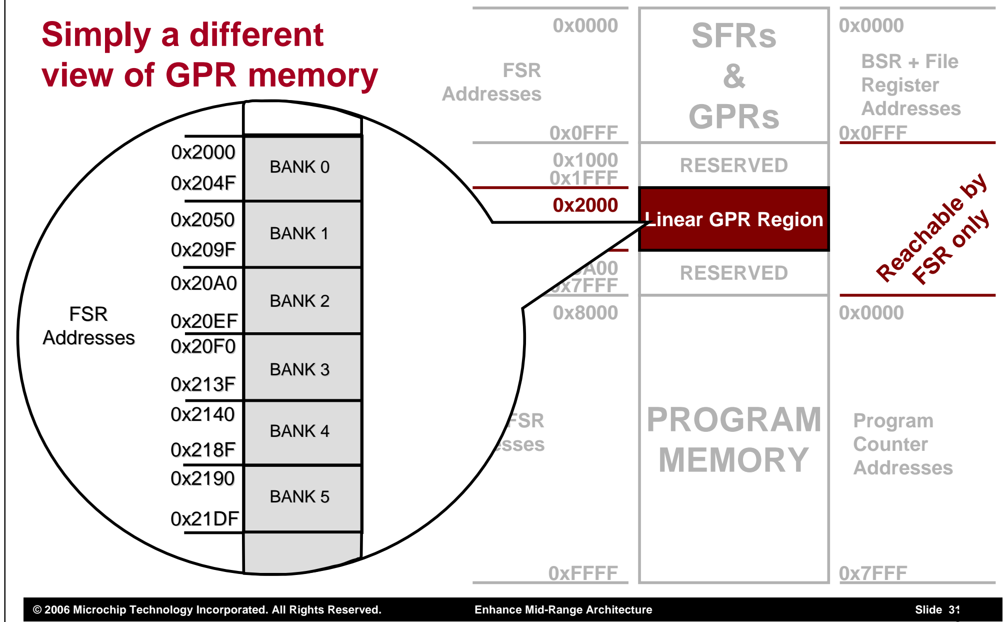
- Access both data and program memory space
- One data pointer for ALL memory
- Supported by new instructions



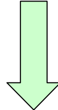
© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 30

These new FSRs along with the new associated instruction we discussed earlier can access nearly ALL the memory in the device. Constants in ROM can be used as easily as variables in RAM.

Shown on this slide is a representation of the File Select Register Memory Map or in other words, memory accessible by any of the two 16-bit File Select Registers. On the left side of the graphic are hexadecimal representations of the memory address locations in both data then program memory spaces. On the right side, these are the addresses accessible by both the bank select register for data memory and the program counter addresses for the program memory. Note in particular the region labeled reachable by the File Select Register only.



This region is called the Linear General Purpose Ram Region. The linear GPR region is simply different view into the GPR area of memory we talked about earlier. Essentially, the GPR space is shadowed into the FSR so there are no gaps between the end of one GPR bank and the beginning of the second GPR bank omitting the core and peripheral special function registers. In this way, all special function registers can be left un altered when accessing arrays of General Purpose RAM that cross from one Bank into another.



Accessing Data Memory with the FSRs

FSRx HIGH

0	0	0	0	0	0	0	0

FSRx LOW

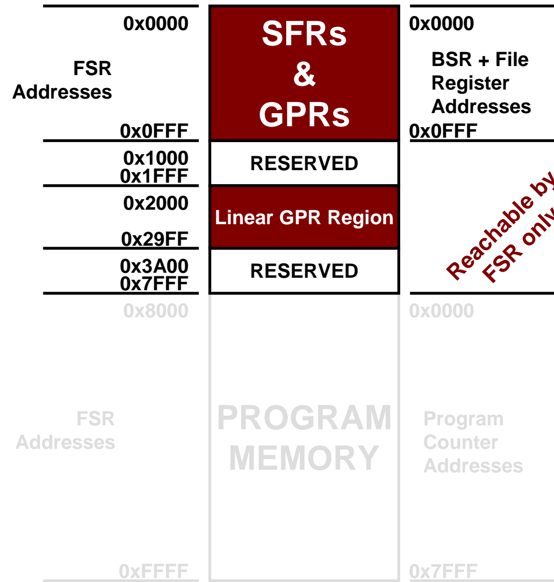
0	0	0	0	0	0	0	0

FSRx HIGH

0	1	1	1	1	1	1	1

FSRx LOW

1	1	1	1	1	1	1	1



© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 32

To differentiate between data and program memory addresses, the Most Significant bit of the FSR High register selected is set or cleared. Shown on the right of this slide are the FSR value to access all data memory including the linear GPR registers I described earlier.

MICROCHIP
WebSeminars

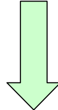
File Select Registers (FSR)

FSRx HIGH

1	0	0	0	0	0	0	0

FSRx LOW

0	0	0	0	0	0	0	0

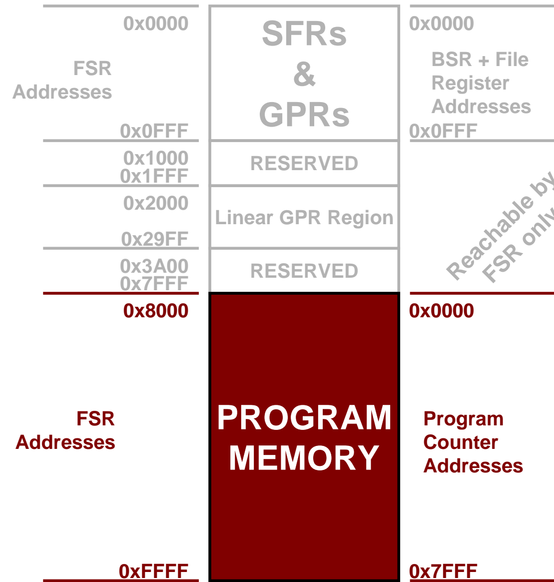


FSRx HIGH

1	1	1	1	1	1	1	1

FSRx LOW

1	1	1	1	1	1	1	1



© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 33

Conversely, to access the program memory space, the Most Significant bit of the FSR High register is set as shown here.

MICROCHIP
WebSeminars

Enhanced Mid-Range Stack

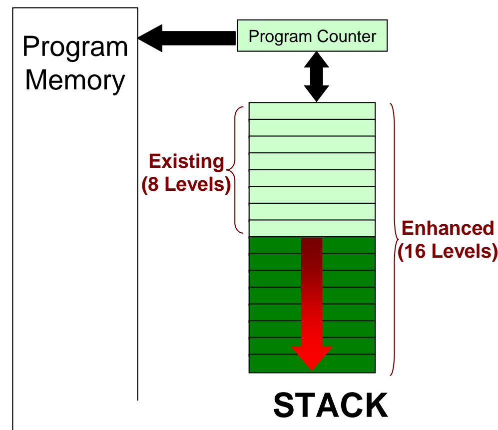
© 2006 Microchip Technology Incorporated. All Rights Reserved.

Let's now discuss enhancements made to the stack…

MICROCHIP WebSeminars

Expanded Hardware Stack

- Extended to 16 Levels
- Increased function/subroutine call depth
- Optional Overflow/Underflow RESET
- User Access



© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 35

As I mentioned toward the beginning of this presentation, the stack has been expanded to 16 levels from 8 levels on the existing Mid-Range architecture. This provides increased interrupt or subroutine call depth. Additionally, the enhanced Mid-Range architecture now features optional overflow/underflow reset capabilities. If the RESET is enabled and either an underflow or overflow condition occurs, the program counter will immediately go to program memory address zero and one of two new reset condition flags in the Power Control Register will identify which condition has occurred. If the RESET is not enabled the stack will simply operate as it did in the existing Mid-Range architecture.

The user also has full access to all 16-levels via three new registers, stack pointer, top of stack high and top of stack low. In this way the user can modify the stack for such advanced techniques as changing the return address following an interrupt or to maintain independent stacks for different operations.

Migrating to the Enhanced Mid-Range Architecture

© 2006 Microchip Technology Incorporated. All Rights Reserved.

Finally, let's discuss migrating existing code to the enhance Mid-Range architecture…

Migration

- **Considerations:**
  - Interrupts
  - Indirect Memory
  - Paging
  - Banking

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 37

Now that we have a basic understanding of the new architecture, let's discuss what should be considered when migrating existing designs to the enhanced Mid-Range architecture. We can simplify migration into four basic areas: interrupts, indirect memory and paging and banking.

Interrupts

MICROCHIP WebSeminars

- **RETFIE works a little different**
  - **Automatically restores context registers**

Address	Register	Function
0x00	INDF0	Indirect Register 0
0x01	INDF1	Indirect Register 1
0x02	PCL	Program Counter Low
0x03	STATUS	Status Register
0x04	FSR0 Low	File Select Register 0 Low Byte
0x05	FSR0 High	File Select Register 0 High Byte
0x06	FSR1 Low	File Select Register 1 Low Byte
0x07	FSR1 High	File Select Register 1 High Byte
0x08	BSR	Bank Select Register
0x09	WREG	Working Register
0x0A	PCLATH	Program Counter Latch High
0x0B	INTCON	Interrupt Control Register

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 38

Referring again to the core registers automatically save on any instruction, the return from interrupt enable instruction now works a little bit different. The OLD RETFIE instruction simply did a return and set GIE. Now all pre-interrupt context is automatically restored.

Interrupts

MICROCHIP WebSeminars

- RETFIE works a little different
  - Automatically restores context registers
- Remove old core context save/restore algorithm
  - If not, this will not affect your code other than wasting clock cycles

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 39

If your application uses the existing method of saving this context in software, this code should be removed from firmware since the peripherals causing the interrupt may now be a bit different.

Indirect Memory

MICROCHIP WebSeminars

- IRP bit in STATUS Register is gone
- Accessing > 256 bytes requires an update to FSRxH register
- **Fastest method** → update FSRxH register
  - Requires modifying W
  - i.e. MOVLW and MOVWF
- BANKISEL is **portable**
  - Performs 8 bcf's or bsf's (i.e. 8 TCY)
  - Does not preserve W

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 40

In the old days, the IRP bit allowed access to the second 256 bytes of data memory. This bit no longer exists in the new architecture; therefore, existing code using this method will need to be changed. The fastest method involves updating the FSR HIGH register which requires a modification to the working register using Move literal to working and then the move working to FSR HIGH .

Or you can do many bit sets or clears. If your code currently uses the BANKISEL macro, then no changes should be made since it is portable into the new assembler. The BANKISEL macro performs all the necessary bit sets or clears in 8 instructions and does not preserve the working register. To preserve the working register, simply add two lines of code around the BANKISEL and the firmware will work on both old and new architectures.

Paging and Banking

MICROCHIP WebSeminars

- **PAGESEL and BANKSEL macros are portable**
  - **Automatically uses MOVLP and MOVLB**

Otherwise

- **Paging**
  - **Update all PCLATH code**
  - **Assure 7 bit data in PCLATH**
- **Banking**
  - **Replace writes to STATUS with writes to BSR**

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 41

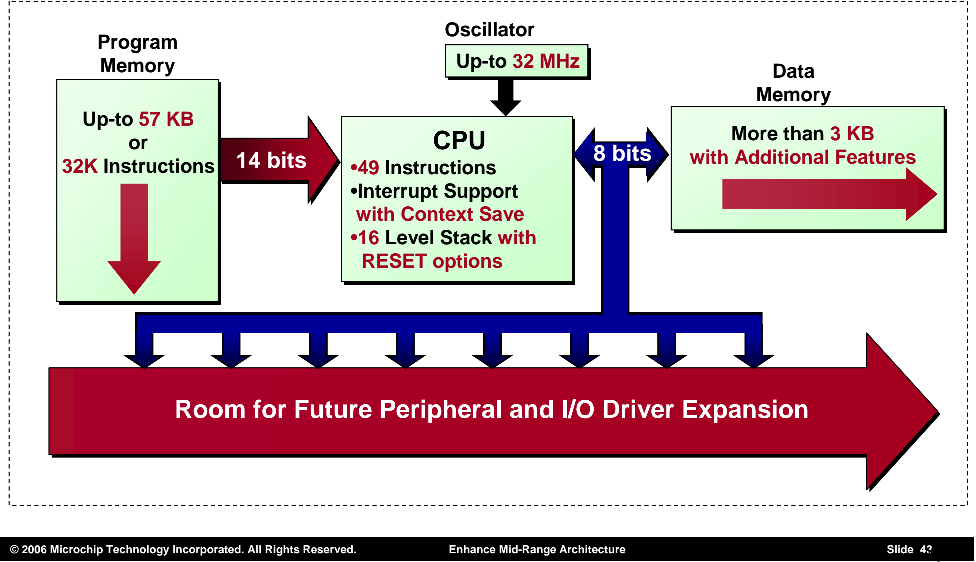
How about paging and banking? If your existing code uses the PAGESEL and BANKSEL macros, there is no need to change anything in firmware since these macros are portable and existing code is automatically converted by the new assembler using the MOVLP and MOVLB instructions in the background. Otherwise, updates to the PCLATH code using the MOVLP and replacing writes to the status register with the MOVLB instruction will be needed.

MICROCHIP
WebSeminars

Summary

© 2006 Microchip Technology Incorporated. All Rights Reserved.

Let's summarize what was discussed in the preceding web seminar…



The enhanced Mid-Range architecture takes a good thing and makes it better. Both program and data memories have been dramatically expanding optimizing the architecture for advanced firmware applications including those programmed in C. 14 new instructions minimize code overhead optimizing instruction execution time. The completely accessible hardware stack with reset capabilities has been expanded to 16 levels. With more room for future peripheral expansion, more memory and 8 MIPS operation, the enhanced mid-range PIC® microcontroller takes the best features of the existing architecture and improves upon them.

Resources

MICROCHIP WebSeminars

- **Enhanced Mid-Range Architecture landing page**

  www.microchip.com/enhanced

  - **PIC16F1XXX Software Migration Document**
  - **PIC16F193X Data Sheet**
  - **PIC16F193X Product Brief**
  - **PIC16F193X Programming Specification**
  - **Enhanced Mid-Range Training Materials**

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhance Mid-Range Architecture Slide 44

For more information on the new enhanced 8-bit Mid-Range PIC® Architecture refer to the resources listed here. Visit the enhanced architecture page at www.microchip.com/enhanced. Here you will find a quick link to such resources as application notes, migration documentation and relevant data sheets for up coming products. The PIC16F1XXX software migration document covers much of what was discussed in the preceding webseminar only in more detail. The soon-to-be-released PIC16F193X family data sheet and collateral resources viewers may find helpful. For future reference all PIC Microcontrollers incorporating the new enhanced Mid-Range core will be designated with the PIC16F1XXX nomenclature. Check back often as new material will be added on a regular basis.

MICROCHIP
WebSeminars

Thank You!

© 2006 Microchip Technology Incorporated. All Rights Reserved. Enhanced Mid-Range Architecture Slide 45

My name is Marc McComb, and I thank you for viewing this web seminar.

Trademarks

The Microchip name and logo, the Microchip logo, Accuron, dsPIC, KeeLoq, KeeLoq logo, MPLAB, PIC, PICmicro, PICSTART, rfPIC, SmartShunt and UNI/O are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, Linear Active Thermistor, MXDEV, MXLAB, SEEVAL, SmartSensor and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Analog-for-the-Digital Age, Application Maestro, CodeGuard, dsPICDEM, dsPICDEM.net, dsPICworks, dsSPEAK, ECAN, ECONOMONITOR, FanSense, In-Circuit Serial Programming, ICSP, ICEPIC, Mindi, MiWi, MPASM, MPLAB Certified logo, MPLIB, MPLINK, mTouch, PICkit, PICDEM, PICDEM.net, PICtail, PIC32 logo, PowerCal, PowerInfo, PowerMate, PowerTool, REAL ICE, rfLAB, Select Mode, Total Endurance, WiperLock and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2008, Microchip Technology Incorporated, All Rights Reserved.