# CHAPTER 2
ADDRESS SPACE

## 2.1 INTRODUCTION

Four address spaces are available for the Z8® MCU:

- The Z8 Standard Register File contains addresses for peripheral, control, all general-purpose, and all I/O port registers. This is the default register file specification.
- The Z8 Expanded Register File (ERF) contains addresses for control and data registers for additional peripherals/features.
- Z8 External Program Memory contains addresses for all memory locations having executable code and/or data.
- Z8 External Data Memory contains addresses for all memory locations that hold data only, whether internal or external.

## 2.2 Z8 MCU STANDARD REGISTER FILE

The Z8 Standard Register File totals up to 256 consecutive bytes (Registers). The register file consists of 4 I/O ports (00H-03H), 236 General-Purpose Registers (04H-EFH), and 16 control registers (F0H-FFH). Table 2-1 shows the layout of the register file, including register names, locations, and identifiers.

**Table 2-1. Z8 Standard Register File**

| Hex Address | Register Description | Register Identifier |
|---|---|---|
| FF | Stack Pointer Low Byte | SPL |
| FE | Stack Pointer High Byte | SPH |
| FD | Register Pointer | RP |
| FC | Program Control Flags | FLAGS |
| FB | Interrupt Mask Register | IMR |
| FA | Interrupt Request Register | IRQ |
| F9 | Interrupt Priority Register | IPR |
| F8 | Port 0-1 Mode Register | P01M |
| F7 | Port 3 Mode Register | P3M |
| F6 | Port 2 Mode Register | P2M |
| F5 | T0 Prescaler | PRE0 |
| F4 | Timer/Counter 0 | T0 |
| F3 | T1 Prescaler | PRE1 |
| F2 | Timer/Counter 1 | T1 |
| F1 | Timer Mode | TMR |

**Table 2-1. Z8 Standard Register File**

| Hex Address | Register Description | Register Identifier |
|---|---|---|
| F0 | Serial I/O | SIO |
| EF | | R239 |
| . | General-Purpose | . |
| . | Registers (GPR) | . |
| . | | . |
| 04 | | R4 |
| 03 | Port 3 | P3 |
| 02 | Port 2 | P2 |
| 01 | Port 1 | P1 |
| 00 | Port 0 | P0 |

## 2.2 Z8 MCU STANDARD REGISTER FILE (Continued)

Registers can be accessed as either 8-bit or 16-bit registers using Direct, Indirect, or Indexed Addressing. All 236 general-purpose registers can be referenced or modified by any instruction that accesses an 8-bit register, without the need for special instructions. Registers accessed as 16 bits are treated as even-odd register pairs (there are 118 valid pairs). In this case, the data's Most Significant Byte (MSB) is stored in the even numbered register, while the Least Significant Byte (LSB) goes into the next higher odd numbered register (Figure 2-1).

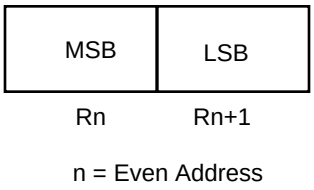


**Figure 2-1. 16-Bit Register Addressing**

By using a logical instruction and a mask, individual bits within registers can be accessed for bit set, bit clear, bit complement, or bit test operations. For example, the instruction AND R15, MASK performs a bit clear operation. Figure 2-2 shows this example.

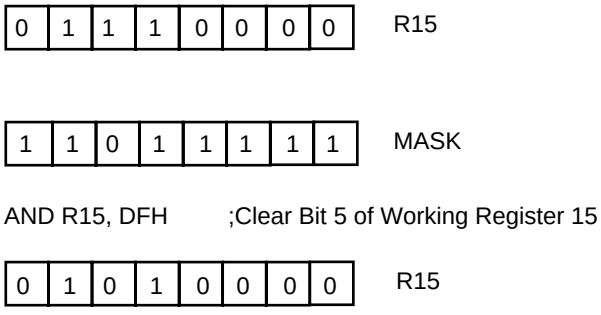


**Figure 2-2. Accessing Individual Bits (Example)**

When instructions are executed, registers are read when defined as sources and written when defined as destinations. All General-Purpose Registers function as accumulators, address pointers, index registers, stack areas, or scratch pad memory.

### 2.2.1 General-Purpose Registers

General-Purpose Registers (GPR) are undefined after the device is powered up. The registers keep their last value after any reset, as long as the reset occurs in the $V_{CC}$ voltage-specified operating range. It will not keep its last state from a $V_{LV}$ reset if $V_{CC}$ drops below 1.8v.

**Note:** Registers in Bank E0-EF may only be accessed through the working register and indirect addressing modes. Direct access cannot be used because the 4-bit working register address mode already uses the format [E | dst], where dst represents the working register number from 0H to FH.

### 2.2.2 RAM Protect

The upper portion of the register file address space 80H to EFH (excluding the control registers) may be protected from reading and writing. The RAM Protect bit option is mask-programmable and is selected by the customer when the ROM code is submitted. After the mask option is selected, the user activates this feature from the internal ROM code to turn off/on the RAM Protect by loading either a 0 or 1 into the IMR register, bit D6. A 1 in D6 enables RAM Protect. Only devices that use registers 80H to EFH offer this feature.

### 2.2.3 Working Register Groups

Z8® instructions can access 8-bit registers and register pairs (16-bit words) using either 4-bit or 8-bit address fields. 8-bit address fields refer to the actual address of the register. For example, Register 58H is accessed by calling upon its 8-bit binary equivalent, 01011000 (58H).

With 4-bit addressing, the register file is logically divided into 16 Working Register Groups of 16 registers each, as shown in Table 2-2. These 16 registers are known as Working Registers. A Register Pointer (one of the control registers, FDH) contains the base address of the active Working Register Group. The high nibble of the Register Pointer determines the current Working Register Group.

When accessing one of the Working Registers, the 4-bit address of the Working Register is combined within the upper four bits (high nibble) of the Register Pointer, thus forming the 8-bit actual address. Figure 2-3 illustrates this operation. Since working registers are typically specified by short format instructions, there are fewer bytes of code needed, which reduces execution time. In addition, when processing interrupts or changing tasks, the Register Pointer speeds context switching. A special Set Register Pointer (SRP) instruction sets the contents of the Register Pointer.

**Table 2-2. Working Register Groups**

| Register Pointer (FDH) High Nibble | Working Register Group (HEX) | Actual Registers (HEX) |
|---|---|---|
| 1111(B) | F | F0–FF |
| 1110(B) | E | E0–EF |
| 1101(B) | D | D0–DF |
| 1100(B) | C | C0–CF |
| 1011(B) | B | B0–BF |
| 1010(B) | A | A0–AF |
| 1001(B) | 9 | 90–9F |
| 1000(B) | 8 | 80–8F |
| 0111(B) | 7 | 70–7F |
| 0110(B) | 6 | 60–6F |
| 0101(B) | 5 | 50–5F |
| 0100(B) | 4 | 40–4F |
| 0011(B) | 3 | 30–3F |
| 0010(B) | 2 | 20–2F |
| 0001(B) | 1 | 10–1F |
| 0000(B) | 0 | 00–0F |

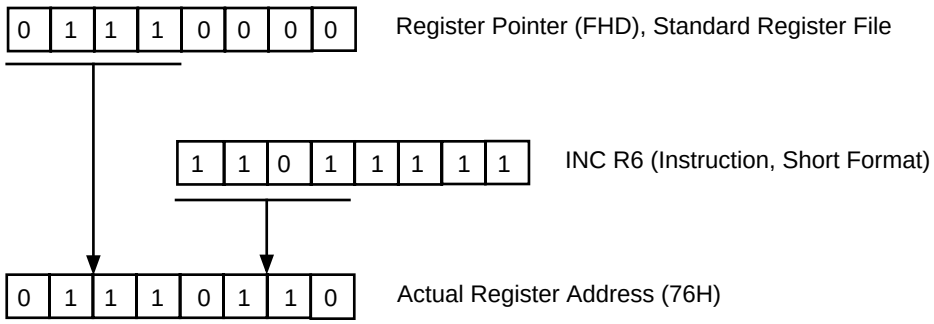


**Figure 2-3. Working Register Addressing Examples**

## 2.2 Z8 MCU STANDARD REGISTER FILE (Continued)

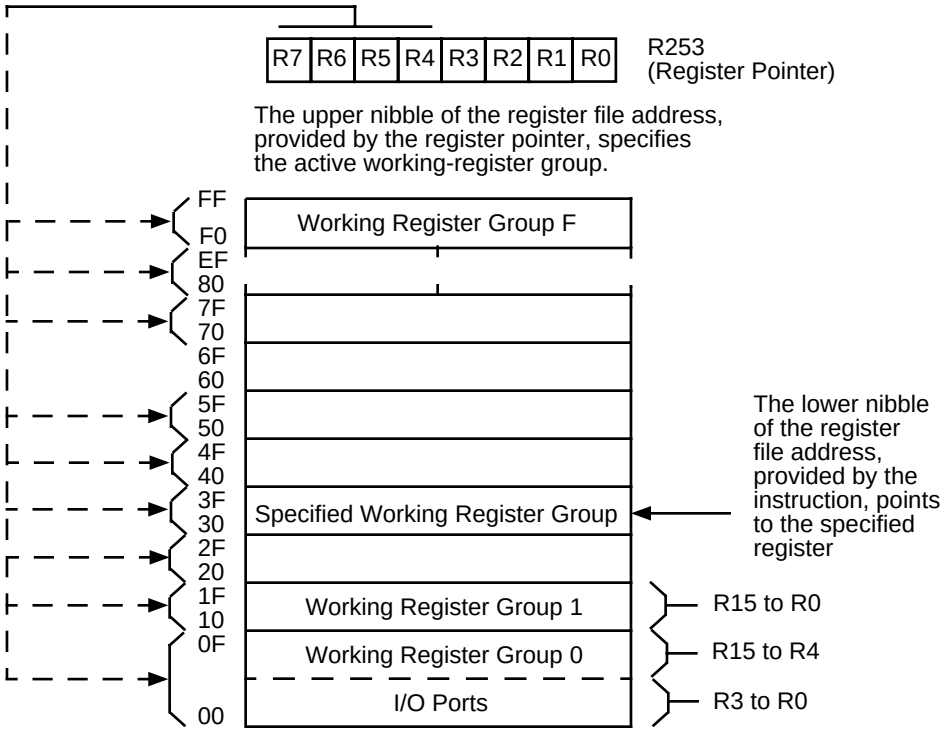


**Figure 2-4. Register Pointer**

**Note:** The full register file is shown. Please refer to the selected device product specification for actual file size.

### 2.2.4 Error Conditions

Registers in the Z8® Standard Register File must be correctly used because certain conditions produce inconsistent results and should be avoided.

- Registers F3H and F5H-F9H are write-only registers. If an attempt is made to read these registers, FFH is returned. Reading any write-only register will return FFH.
- When register FDH (Register Pointer) is read, the least significant four bits (lower nibble) will indicate the current Expanded Register File Bank. (Example: 0000 indicates the Standard Register File, while 1010 indicates Expanded Register File Bank A.)
- When Ports 0 and 1 are defined as address outputs, registers 00H and 01H will return 1s in each address bit location when read.
- Writing to bits that are defined as timer output, serial output, or handshake output will have no effect.
- The Z8 instruction DJNZ uses any general-purpose working register as a counter.
- Logical instructions such as OR and AND require that the current contents of the operand be read. They therefore will not function properly on write-only registers.
- The WDTMR register must be written within the first 60 internal system clocks (SCLK) of operation after a reset.

## 2.3 Z8 EXPANDED REGISTER FILE

The standard register file of the Z8® has been expanded to form 16 Expanded Register File (ERF) Banks (Figure 2-5). Each ERF Bank consists of up to 256 registers (the same amount as in the Standard Register File) that can then be divided into 16 Working Register Groups. This expansion allows for access to additional feature/peripheral control and data registers.

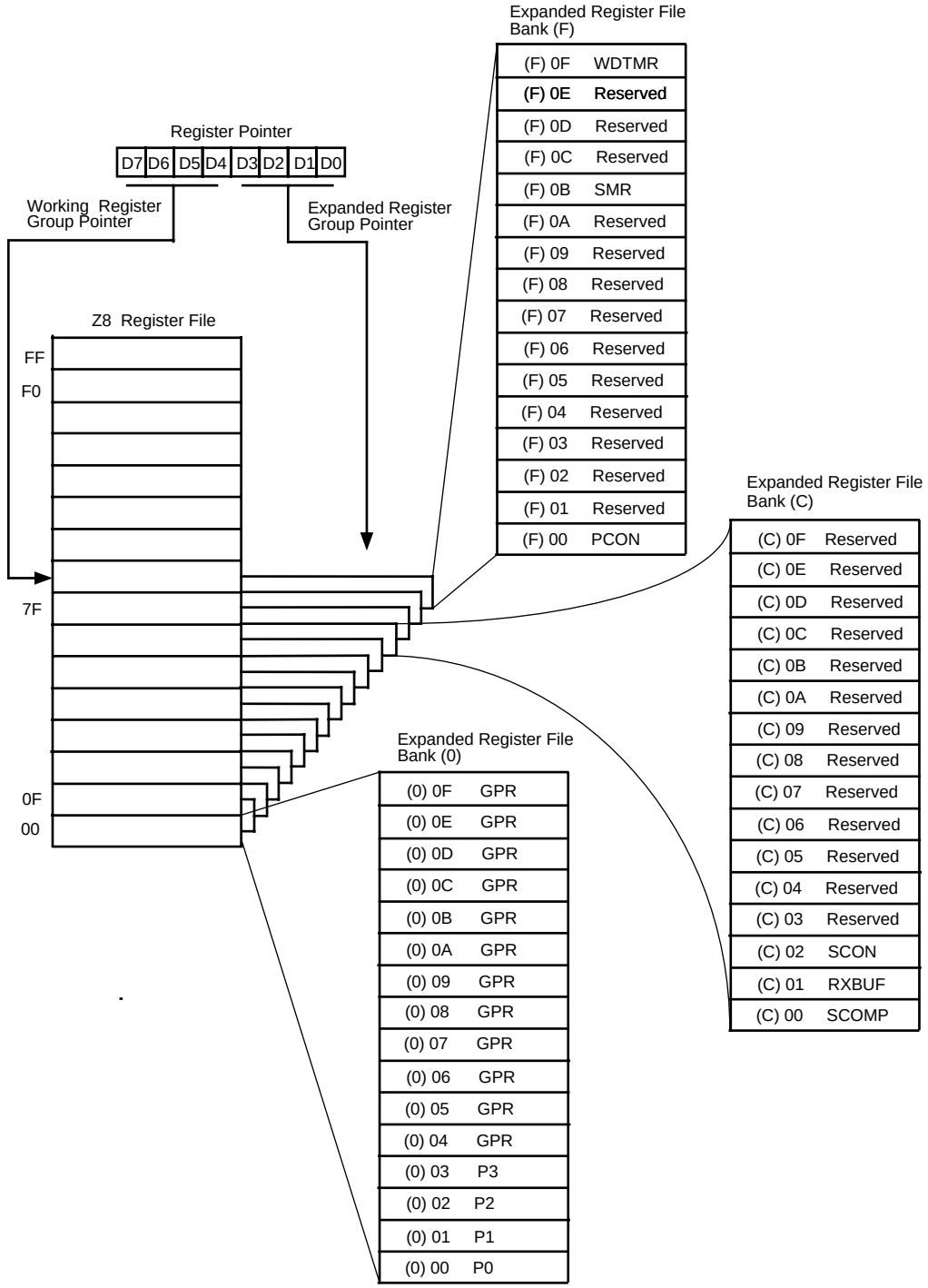


**Figure 2-5. Expanded Register File Architecture**

**Note:** The fully implemented register file is shown. Please refer to the specific product specification for actual register file architecture implemented.

## 2.3 Z8 EXPANDED REGISTER FILE (Continued)

Currently, three out of the possible sixteen Z8® ERF Banks have been implemented. ERF Bank 0, also known as the Z8 Standard Register File, has all 256 bytes defined (Figure 2-1). Only Working Register Group 0 (register addresses 00H to 0FH) have been defined for ERF Bank C and ERF Bank F (Table 2-4). All other working register groups in ERF Banks C and F, as well as the remaining thirteen ERF Banks, are not implemented. All are reserved for future use.

When an ERF Bank is selected, register addresses 00H to 0FH access those sixteen ERF Bank registers – in effect replacing the first sixteen locations of the Z8 Standard Register File.

For example, if ERF Bank C is selected, the Z8 Standard Registers 00H through 0FH are no longer accessible. Registers 00H through 0FH are now the 16 registers from ERF Bank C, Working Register Group 0. No other Z8 Standard Registers are effected since only Working Register Group 0 is implemented in ERF Bank C.

Access to the ERF is accomplished through the Register Pointer (FDH). The lower nibble of the Register Pointer determines the ERF Bank while the upper nibble determines the Working Register Group within the register file (Figure 2-6).

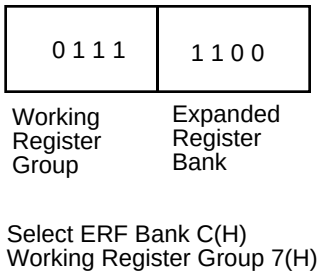


**Figure 2-6. Register Pointer (FDH) Example**

The value of the lower nibble in the Register Pointer (FDH) corresponds to the ERF Bank identification. Table 2.3 shows the lower nibble value and the register file assigned to it.

**Table 2-3. ERF Bank Address**

| Register Pointer (FDH) Low Nibble | Hex | Register File |
|---|---|---|
| 0000(B) | 0 | Z8® Standard Register File * |
| 0001(B) | 1 | Expanded Register File Bank 1 |
| 0010(B) | 2 | Expanded Register File Bank 2 |
| 0011(B) | 3 | Expanded Register File Bank 3 |
| 0100(B) | 4 | Expanded Register File Bank 4 |
| 0101(B) | 5 | Expanded Register File Bank 5 |
| 0110(B) | 6 | Expanded Register File Bank 6 |
| 0111(B) | 7 | Expanded Register File Bank 7 |
| 1000(B) | 8 | Expanded Register File Bank 8 |
| 1001(B) | 9 | Expanded Register File Bank 9 |
| 1010(B) | A | Expanded Register File Bank A |
| 1011(B) | B | Expanded Register File Bank B |
| 1100(B) | C | Expanded Register File Bank C |
| 1101(B) | D | Expanded Register File Bank D |
| 1110(B) | E | Expanded Register File Bank E |
| 1111(B) | F | Expanded Register File Bank F |

**Note:** The Z8 Standard Register File is equivalent to Expanded Register File Bank 0.

The upper nibble of the register pointer selects which group of 16 bytes in the Register File, out of the full 256, will be accessed as working registers.

**For example:**
(See Figure 2-4)

```
R253 RP = 00H    ;ERF Bank 0, Working Reg. Group 0.
                 R0 = Port 0 = 00H
                 R1 = Port 1 = 01H
                 R2 = Port 2 = 02H
                 R3 = Port 3 = 03H
                 R11 = GPR 0BH
                 R15 = GPR 0FH
If:
R253 RP = 0FH    ;ERF Bank F, Working Reg. Group 0.
                 R0 = PCON = 00H
                 R1 = Reserved = 01H
                 R2 = Reserved = 02H
                 R11 = SMR = 0BH
                 R15 = WDTMR = 0FH
If:
R253 RP = FFH    ;ERF Bank F, Working Reg. Group F.
                 00H = PCON
                 R0 = SI0        01H= Reserved
                 R1 = TMR        02H= Reserved
                                 ...
                 R2 = T1         0BH = SMR
                                 ...
                 R15 = SPL       0FH = WDTMR
```

Note that since enabling an ERF Bank (C or F) only changes register addresses 00H to 0FH, the working register pointer can be used to access either the selected ERF Bank (Bank C or F, Working Register Group 0) or the Z8 Standard Register File (ERF Bank 0, Working Register Groups 1 through F).

**Note:** When an ERF Bank other than Bank 0 is enabled, the first 16 bytes of the Z8 Standard Register File (I/O ports 0 to 3, Groups 4 to F) are no longer accessible (the selected ERF Bank, Registers 00H to 0FH are accessed instead). It is important to re-initialize the Register Pointer to enable ERF Bank 0 when these registers are required for use.

The SPI register is mapped into ERF Bank C. Access is easily done using the following example:

```
LD      RP, #0CH    ;Select ERF Bank C working
                    ;register group 0 for access.
LD      R2,#xx      ;access SCON
LD      R1, #xx     ;access RXBUF
LD      RP, #00H    ;Select ERF Bank 0 so I/O ports
                    ;are again accessible.
```

**Table 2-4. Z8 Expanded Register File Bank Layout**

| Expanded Register File Bank | ERF |
|---|---|
| F(H) | PCON, SMR, WDT, (00H, 0BH, 0FH), Working Register Group 0 only implemented. |
| E(H) | Not Implemented (Reserved) |
| D(H) | Not Implemented (Reserved) |
| C(H) | SPI Registers: SCOMP, RXBUF, SCON (00H, 01H, 02H), Working Register Group 0 only implemented. |
| B(H) | Not Implemented (Reserved) |
| A(H) | Not Implemented (Reserved) |
| 9(H) | Not Implemented (Reserved) |
| 8(H) | Not Implemented (Reserved) |
| 7(H) | Not Implemented (Reserved) |
| 6(H) | Not Implemented (Reserved) |
| 5(H) | Not Implemented (Reserved) |
| 4(H) | Not Implemented (Reserved) |
| 3(H) | Not Implemented (Reserved) |
| 2(H) | Not Implemented (Reserved) |
| 1(H) | Not Implemented (Reserved) |
| 0(H) | Z8 Ports 0, 1, 2, 3, and General-Purpose Registers 04H to EFH, and control registers F0H to FFH. |

Please refer to the specific product specification to determine the above registers are implemented.

## 2.4 Z8 CONTROL AND PERIPHERAL REGISTERS

### 2.4.1 Standard Z8 Registers

The standard Z8® control registers govern the operation of the CPU. Any instruction which references the register file can access these control registers. Available control registers are:

- Interrupt Priority Register (IPR)
- Interrupt Mask Register (IMR)
- Interrupt Request Register (IRQ)
- Program Control Flags (FLAGS)
- Register Pointer (RP)
- Stack Pointer High-Byte (SPH)
- Stack Pointer Low-Byte (SPL)

The Z8 uses a 16-bit Program Counter (PC) to determine the sequence of current program instructions. The PC is not an addressable register.

Peripheral registers are used to transfer data, configure the operating mode, and control the operation of the on-chip peripherals. Any instruction that references the register file can access the peripheral registers. The peripheral registers are:

- Serial I/O (SIO)
- Timer Mode (TMR)
- Timer/Counter 0 (T0)
- T0 Prescaler (PRE0)
- Timer/Counter 1 (T1)
- T1 Prescaler (PRE1)
- Port 0–1 Mode (P01M)
- Port 2 Mode (P2M)
- Port 3 Mode (P3M)

In addition, the four port registers (P0–P3) are considered to be peripheral registers.

### 2.4.2 Expanded Z8 Registers

The expanded Z8 control registers govern the operation of additional features or peripherals. Any instruction which references the register file can access these registers.

The ERF contains the control registers for WDT, Port Control, Serial Peripheral Interface (SPI), and the SMR functions. Figure 2-4 shows the layout of the Register Banks in the ERF. Register Bank C in the ERF consists of the registers for the SPI. Table 2-5 shows the registers within ERF Bank C, Working Register Group 0.

**Table 2-5. Expanded Register File Register Bank C, WR Group 0**

| Register | Register Function | Working Register |
|---|---|---|
| F | Reserved | R15 |
| E | Reserved | R14 |
| D | Reserved | R13 |
| C | Reserved | R12 |
| B | Reserved | R11 |
| A | Reserved | R10 |
| 9 | Reserved | R9 |
| 8 | Reserved | R8 |
| 7 | Reserved | R7 |
| 6 | Reserved | R6 |
| 5 | Reserved | R5 |
| 4 | Reserved | R4 |
| 3 | Reserved | R3 |
| 2 | SPI Control (SCON) | R2 |
| 1 | SPI Tx/Rx Data (Roxburgh) | R1 |
| 0 | SPI Compare (SCOMP) | R0 |

Working Register Group 0 in ERF Bank 0 consists of the registers for Z8 General-Purpose Registers and ports. Table 2-6 shows the registers within this group.

**Table 2-6. Expanded Register File Bank 0, WR Group 0**

| Register | Register Function | Working Register |
|---|---|---|
| F | General-Purpose Register | R15 |
| E | General-Purpose Register | R14 |
| D | General-Purpose Register | R13 |
| C | General-Purpose Register | R12 |
| B | General-Purpose Register | R11 |
| A | General-Purpose Register | R10 |
| 9 | General-Purpose Register | R9 |
| 8 | General-Purpose Register | R8 |
| 7 | General-Purpose Register | R7 |
| 6 | General-Purpose Register | R6 |
| 5 | General-Purpose Register | R5 |
| 4 | General-Purpose Register | R4 |
| 3 | Port 3 | R3 |
| 2 | Port 2 | R2 |
| 1 | Port 1 | R1 |
| 0 | Port 0 | R0 |

Working Register Group 0 in ERF Bank F consists of the control registers for STOP mode, WDT, and port control. Table 2-7 shows the registers within this group.

**Table 2-7. Expanded Register File Bank F, WR Group 0**

| Register | Register Function | Working Register |
|---|---|---|
| F | WDTMR | R15 |
| E | Reserved | R14 |
| D | Reserved | R13 |
| C | Reserved | R12 |
| B | SMR | R11 |
| A | Reserved | R10 |
| 9 | Reserved | R9 |
| 8 | Reserved | R8 |
| 7 | Reserved | R7 |
| 6 | Reserved | R6 |
| 5 | Reserved | R5 |
| 4 | Reserved | R4 |
| 3 | Reserved | R3 |
| 2 | Reserved | R2 |
| 1 | Reserved | R1 |
| 0 | PCON | R0 |

The functions and applications of the control and peripheral registers are described in subsequent sections of this manual.

## 2.5 PROGRAM MEMORY

The first 12 bytes of Program Memory are reserved for the interrupt vectors (Figure 2-7). These locations contain six 16-bit vectors that correspond to the six available interrupts. Address 12 up to the maximum ROM address consists of on-chip mask-programmable ROM. See the product data sheet for the exact program, data, register memory size, and address range available. At addresses outside the internal ROM, the Z8® executes external program memory fetches through Port 0 and Port 1 in Address/Data mode for devices with Port 0 and Port 1 featured. Otherwise, the program counter will continue to execute NOPs up to address FFFFH, roll over to 0000H, and continue to fetch executable code (Figure 2-7).

The internal program memory is one-time programmable (OTP) or mask programmable dependent on the specific device. ***A ROM protect feature prevents "dumping" of the ROM contents by inhibiting execution of the LDC, LDCI, LDE, and LDEI instructions to Program Memory in all modes. ROM look-up tables cannot be used with this feature.***

The ROM Protect option is mask-programmable, to be selected by the customer when the ROM code is submitted. For the OTP ROM, the ROM Protect option is an OTP programming option.

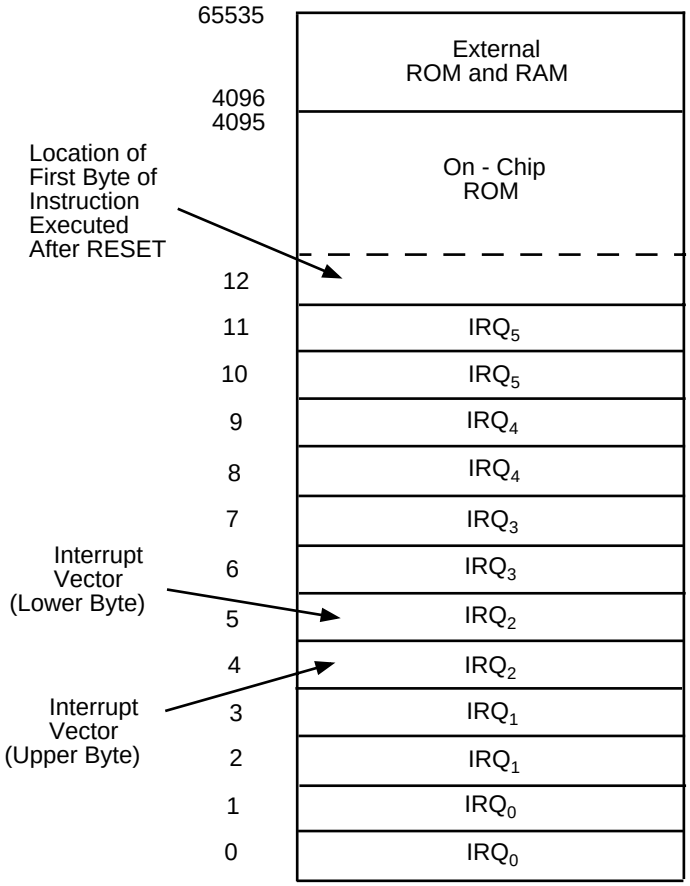


**Figure 2-7. Z8 Program Memory Map**

## 2.6 Z8 EXTERNAL MEMORY

The Z8®, in some cases, has the capability to access external program memory with the 16-bit Program Counter. To access external program memory the Z8 offers multiplexed address/data lines (AD7-AD0) on Port 1 and address lines (A15-A8) on Port 0. This feature only applies to devices that offer Port 0 and Port 1. The maximum external address is FFFF. This memory interface is supported by the control lines /AS (Address Strobe), /DS (Data Strobe), and R/W (Read/Write). The origin of the external program memory starts after the last address of the internal ROM. Figure 2-8 shows an example of external program memory for the Z8.

### 2.6.1 External Data Memory (/DM)

The Z8, in some cases, can address up to 60 Kbytes of external data memory beginning at location 4096. External Data Memory may be included with, or separated from, the external Program Memory space. /DM, an optional I/O function that can be programmed to appear on pin P34, is used to distinguish between data and program memory space. The state of the /DM signal is controlled by the type of instruction being executed. An LDC opcode references Program (/DM inactive) Memory, and an LDE instruction references Data (/DM active Low) Memory. The user must configure Port 3 Mode Register (P3M) bits D3 and D4 for this mode.

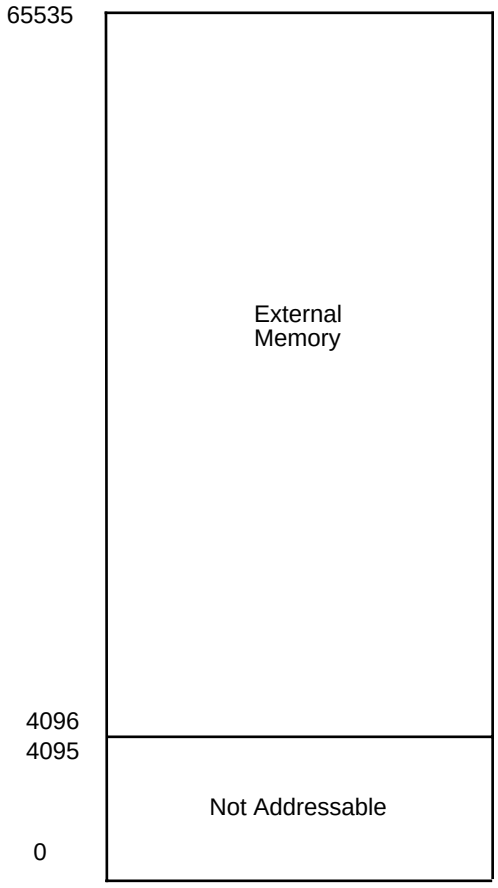


**Figure 2-8. External Memory Map**

**Note:** For additional information on using external memory, see Chapter 10 of this manual. For exact memory addressing options available, see the device product specification.

## 2.7 Z8 STACKS

Stack operations can occur in either the Z8® MCU Standard Register File or external data memory. Under software control, Port 0–1 Mode register (F8H) selects the stack location. Only the General-Purpose Registers can be used for the stack when the internal stack is selected.

The register pair FEH and FFH form the 16-bit Stack Pointer (SP), that is used for all stack operations. The stack address is stored with the MSB in FEH and LSB in FFH (Figure 2-9).

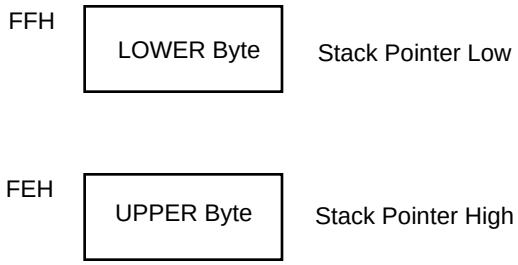


**Figure 2-9. Stack Pointer**

The stack address is decremented prior to a PUSH operation and incremented after a POP operation. The stack address always points to the data stored on the top of the stack. The Z8® stack is a return stack for CALL instructions and interrupts, as well as a data stack.

During a CALL instruction, the contents of the PC are saved on the stack. The PC is restored during a RETURN instruction. Interrupts cause the contents of the PC and Flag registers to be saved on the stack. The IRET instruction restores them (Figure 2-10).

When the Z8 is configured for an internal stack (using the Z8 Standard Register File), register FFH serves as the Stack Pointer. The value in FEH is ignored. FEH can be used as a general-purpose register in this case only.

An overflow or underflow can occur when the stack address is incremented or decremented during normal stack operations. The programmer must prevent this occurrence or unpredictable operation will result.

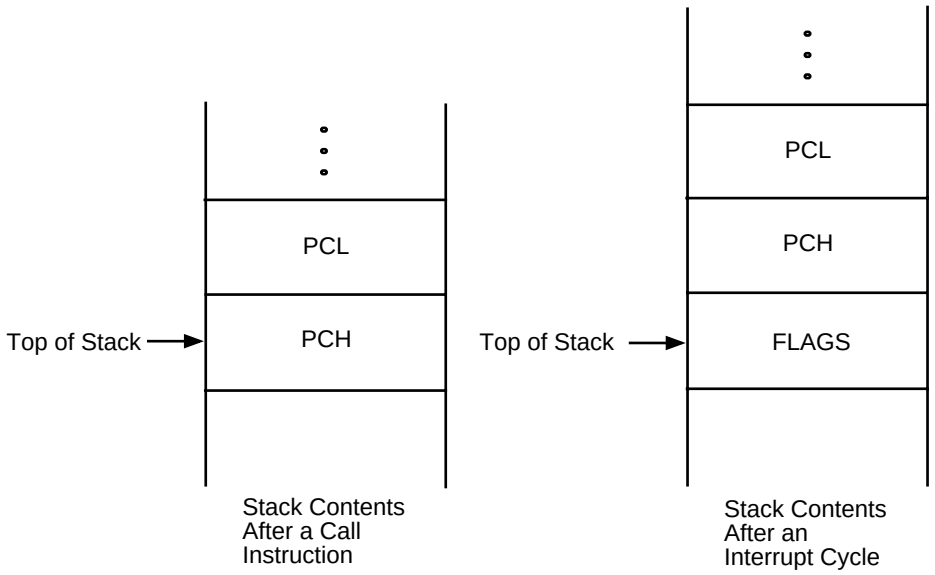


**Figure 2-10. Stack Operations**