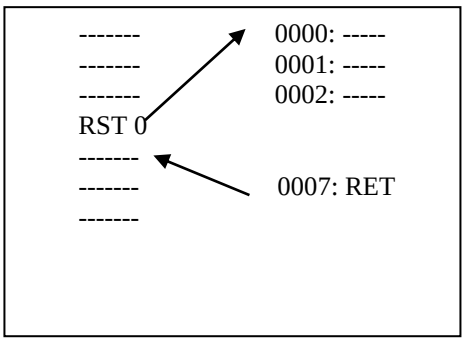
Software Interrupt (Restart Instructions)

The 8085 has eight restart instructions: RST 0, RST 1, RST 2, RST 3, RST 4, RST 5, RST 6, and RST 7. These instructions were originally used in the interrupt system of the 8080. In the 8085, however, the restart instructions represent efficient to call frequently used subroutines.

No.	Instruction	Type	No. of Bytes	Function	Effect
1.	RST 0	Restart	1	PC=0000	None
2.	RST 1	Restart	1	PC=0008	None
3.	RST 2	Restart	1	PC=0010	None
4.	RST 3	Restart	1	PC=0018	None
5.	RST 4	Restart	1	PC=0020	None
6.	RST 5	Restart	1	PC=0028	None
7.	RST 6	Restart	1	PC=0030	None
8.	RST 7	Restart	1	PC=0038	None

What an RST does? The table shows the effect of each restart instruction. As indicated, an RST has the same effect as a call. For instance, the execution of RST0 will begin by pushing the contents of the program counter onto the stack. Then the program branches to address 0000h as shown below. The subroutine located between 0000h and 0007h is carried out with the RET returning the processing to the main program.



A restart instruction is a special kind of call because it branches to a predetermined address. Notice that only 1 byte is required to code a restart instruction. The standard call uses three bytes; therefore, an RST instruction is an efficient way to call frequently used subroutines.

Vectored Calls

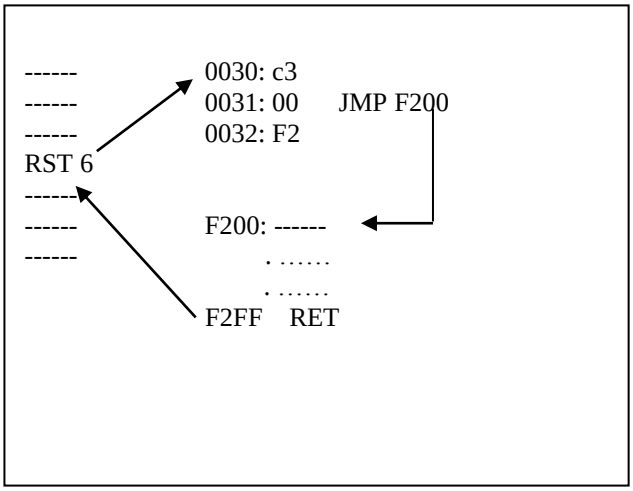
The word vector implies direction. The RST instructions are like vectors because they point to specific locations in memory. The starting address of each restart subroutine is called a vector location. RST 0 points to vector location 0000H, RST 1 points to vector location 0008H, and so on.

Notice that there are only 8 bytes from 0000H to 0007H, 0008H to 000FH, 0010H to 0017H, and so on. Most useful subroutines require a lot more than 8 bytes. For this reason you rarely see subroutines stored in the restart locations. Instead, most programmers use the vector locations for the starting address of longer subroutines.

An example

Suppose a subroutine with 256 bytes is stored at address F200H to F2FFH. If this subroutine is used a great deal, we can call it efficiently by using a RST 6 with a JMP F200H stored at address 0030H The RST 6 takes the program to address 0030H; then the JMP F200H takes the program to the subroutine.

The following figure illustrates the program flow. When the program encounters the RST 6, it branches to address 0030H. Here it finds a JMP F200H. This produces a branch to the subroutine located at F200H. The long subroutine is then executed with the final instruction RET taking the processing back to the main program.



Class Work

Use RST 5 to jump to subroutine to perform $C=B^2+5$

Address	HexCode	Label	Opcode	Operands	Comments
2000 2001 2002	 50 20		LXI	SP,2050	; SP=2050
2003 2004			MVI	B,	; B=
2005			RST5		; PC=2028
2006			RST1		; END
2007		SUBR:	MOV	A,B	; A=B
2008			MOV	C,B	; C=B
2009			DCR	C	; C=C-1
200A		NEXT:	ADD	B	; A=A+B
200B			DCR	C	; C=C-1
200C 200D 200E	 0A 20		JNZ	NEXT	; ; IF Z=0 then PC=200A
200F			ADD	B	; A=A+5
2010			MOV	C,A	; C=A
2011			RET		; END

RST5

0028: C3

0029: 07 } JMP SUBR

002A : 20