

US8051/52

Fast Speed i8051/52 Microcontroller Core

Application Notes

ABSTRACT

The FS8051/52 Application Notes contain the description of the FS8051/52 core architecture. FS8051/52 soft core implements the instruction set and hardware resources of widely used i8051/52 microcontroller. The shortened instruction cycles give the advantages to increase the throughput or minimize the power consumption. The core intended for implementation in Xilinx Virtex™, SpartanII™, Spartan3™, Virtex4™ and Virtex5™ devices.

FEATURES

Key features

- Up to 130 MHz clock frequency (in Virtex2 devices),
- 2, 3, and 4 -cycle instruction periods,
- 970 or 1076 CLB slices, and 1 Block_RAM for i8052 architecture,
- high speed JUMP,CALL,RET instructions and interrupt handling,
- up to 100 user defined Special Function registers can be attached,
- up to 2 Mbytes of outer addressable memory with programmed select impulse width,
- Structure optimized for Xilinx Virtex™, SpartanII™, Spartan3™, Virtex4™ and Virtex5™ FPGA devices,
- fully parametrized core. The initial hardware volume can be minimized down to 500 CLB slices depending on resource and/or instruction set utilization.
- Supports different technologies (Xilinx, Altera, Actel).

Design features

- **High rate of the clock frequency and shorten instruction period.**

High rate of the clock frequency up to 130 MHz (in VirtexII devices) and shorten instruction period gives the advantage of achieving the dramatically high throughput comparing to the standard 8051 microcontroller devices. Each instruction is implemented only for 2, 3, or 4 clock cycles. When the DLL/DCM unit is used in the FPGA device then one can implement the clock signal frequency multiplication by 2, and using the outer clock generator with 65 MHz of frequency, and each instruction will be implemented for 1, 1.5, or 2 clock periods.

The use of synchronous program memory helps to utilize the BlockSelectRAMs as the program storage, and supports the high rate of the clock frequency when the program volume is high. (Analogous cores do not utilize synchronous program memory.)

- **High speed JUMP,CALL,RET instructions and interrupt handling.**

All the Jump -type instructions are implemented only for 3, or 4 instruction cycles, which is rather shorter than by analogous microcontrollers and cores. The interrupt handling speed up increases the sharpness of the core applications.

- **Small hardware volume.**

The bare core without timers and UART occupies only 656 CLB slices which supports the configuration it into the popular and cheap Xilinx SpartanII devices. The deep parametrization of the core helps to minimize the hardware for the cost of unnecessary functionality. For example, if the application does not use MUL, DIV, DA instructions, then the hardware is minimized for the cost of the multiplier, divider, and decimal adjust units.

Performance

The following table illustrates the FS8051/52core performance in Xilinx VIRTEX™ device.

Core configuration	Core 8051	Core 8051	Core 8052	Core 8052
Target device	XCV200E -8	XC2V500-5	XCV200E-8	XC2V500-5
Select Memory	1 Block_RAM	1 Block_RAM	1 Block_RAM	1 Block_RAM
Area	864 Slices	942 Slices	970 Slices	1076 Slices
System clock fmax	80.6 MHz	130 MHz	77.4 MHz	125 MHz

Table 1. Implementation Data – Xilinx VIRTEX-E and VIRTEX-2

Note: additional BlockRAMs are needed which number depends on the program volume.

Signal description

SIGNAL	TYPE	DESCRIPTION
CLK	input	Global clock
GATE0	input	Timer 0 gate input
GATE1	input	Timer 1 gate input
INT0	input	External interrupt 0
INT1	input	External interrupt 1
PORT0I[7:0]	input	Port 0 input
PORT1I[7:0]	input	Port 1 input
PORT2I[7:0]	input	Port 2 input
PORT3I[7:0]	input	Port 3 input
INSTRI[7:0]	input	Data bus from program memory
RST	input	Global reset
RXDI	input	Serial receiver input
SFRDATAI[7:0]	input	Data bus from user SFR's
T0	input	Timer 0 input
T1	input	Timer 1 input
MEMDATAI[7:0]	input	Data bus from external data memory
INSTADDR[15:0]	output	Instruction address bus
PORT0O[7:0]	output	Port 0 output
PORT1O[7:0]	output	Port 1 output
PORT2O[7:0]	output	Port 2 output
PORT3O[7:0]	output	Port 3 output
TXD	output	Serial transmitter output
RXDO	output	Serial receiver output
SFRADDRS[7:0]	output	RAM and SFR's source address bus
SFRADDRD[7:0]	output	RAM and SFR's destination address bus
SFRDATAO[7:0]	output	Data bus to user SFR's
SFROE	output	User SFR's read
SFRWE	output	User SFR's write enable
MEMADDR[23:0]	output	External data memory address bus
MEMDATAO[7:0]	output	Data bus for external data memory
MEMWR	output	External data memory write
MEMRD	output	External data memory read

Table 2. FS8051/52 core signal description.

Instruction Cycles

The following tables give information about the instruction cycles of the FS8051core. Table 3 and Table 4 contain notes for mnemonics used in Instruction set tables. Tables 5-8 show instruction hexadecimal codes, number of bytes and clock periods for each instruction.

Rn	Working register R0-R7
Direct	128 internal RAM locations, Special Function Registers
@Ri	Indirect internal or external RAM location addressed by index register R0 or R1
#data	8-bit constant included in instruction
#data 16	16-bit constant included as bytes 2 and 3 of instruction

Bit	256 software flags, any bit-addressable I/O pin, control or status bit
A	Accumulator

Table 3. Mnemonics on data addressing modes

Addr16	Destination address for LCALL and LJMP may be anywhere within the 64-Kbyte of program memory address space.
Addr11	Destination address for ACALL and AJMP will be within the same 2-Kbyte page of program memory as the first byte of the following instruction.
Rel	SJMP and all conditional jumps include an 8-bit offset byte. Range is +127/-128 bytes relative to the first byte of the following instruction

Table 4. Mnemonics on program addressing modes

Mnemonic	Description	Code	Bytes	Clk periods
ADD A,Rn	Add register to accumulator	28-2F	1	3
ADD A,direct	Add direct byte to accumulator	25	2	3
ADD A,@Ri	Add indirect RAM to accumulator	26-27	1	3
ADD A,#data	Add immediate data to accumulator	24	2	3
ADDC A,Rn	Add register to accumulator with carry flag	38-3F	1	3
ADDC A,direct	Add direct byte to A with carry flag	35	2	3
ADDC A,@Ri	Add indirect RAM to A with carry flag	36-37	1	3
ADDC A,#data	Add immediate data to A with carry flag	34	2	3
SUBB A,Rn	Subtract register from A with borrow	98-9F	1	3
SUBB A,direct	Subtract direct byte from A with borrow	95	2	3
SUBB A,@Ri	Subtract indirect RAM from A with borrow	96-97	1	3
SUBB A,#data	Subtract immediate data from A with borrow	94	2	3
INC A	Increment accumulator	04	1	3
INC Rn	Increment register	08-0F	1	3
INC direct	Increment direct byte	05	2	3
INC @Ri	Increment indirect RAM	06-07	1	3
DEC A	Decrement accumulator	14	1	3
DEC Rn	Decrement register	18-1F	1	3
DEC direct	Decrement direct byte	15	1	3
DEC @Ri	Decrement indirect RAM	16-17	2	3
INC DPTR	Increment data pointer	A3	1	2
MUL A,B	Multiply A and B	A4	1	2
DIV A,B	Divide A by B	84	1	4
DA A	Decimal adjust accumulator	D4	1	2

Table 4. Arithmetic operation instructions

Mnemonic	Description	Code	Bytes	Clk periods
ANL A,Rn	AND register to accumulator	58-5F	1	2
ANL A,direct	AND direct byte to accumulator	55	2	2
ANL A,@Ri	AND indirect RAM to accumulator	56-57	1	2
ANL A,#data	AND immediate data to accumulator	54	2	2
ANL direct,A	AND accumulator to direct byte	52	2	3
ANL direct,#data	AND immediate data to direct byte	53	3	3
ORL A,Rn	OR register to accumulator	48-4F	1	2
ORL A,direct	OR direct byte to accumulator	45	2	2

ORL A,@Ri	OR indirect RAM to accumulator	46-47	1	2
ORL A,#data	OR immediate data to accumulator	44	2	2
ORL direct,A	OR accumulator to direct byte	42	2	3
ORL direct,#data	OR immediate data to direct byte	43	3	3
XRL A,Rn	Exclusive OR register to accumulator	68-6F	1	2
XRL A,direct	Exclusive OR direct byte to accumulator	65	2	2
XRL A,@Ri	Exclusive OR indirect RAM to accumulator	66-67	1	2
XRL A,#data	Exclusive OR immediate data to accumulator	64	2	2
XRL direct,A	Exclusive OR accumulator to direct byte	62	2	3
XRL direct,#data	Exclusive OR immediate data to direct byte	63	3	3
CLR A	Clear accumulator	E4	1	2
CPL A	Complement accumulator	F4	1	2
RL A	Rotate accumulator left	23	1	2
RLC A	Rotate accumulator left through carry	33	1	2
RR A	Rotate accumulator right	03	1	2
RRC A	Rotate accumulator right through carry	13	1	2
SWAP A	Swap nibbles within the accumulator	C4	1	2

Table 5. Logic operation instructions

Mnemonic	Description	Code	Bytes	Clk periods
MOV A,Rn	Move register to accumulator	E8-EF	1	2
MOV A,direct	Move direct byte to accumulator	E5	2	2
MOV A,@Ri	Move indirect RAM to accumulator	E6-E7	1	2
MOV A,#data	Move immediate data to accumulator	74	2	2
MOV Rn,A	Move accumulator to register	F8-FF	1	2
MOV Rn,direct	Move direct byte to register	A8-AF	2	2
MOV Rn,#data	Move immediate data to register	78-7F	2	2
MOV direct,A	Move accumulator to direct byte	F5	2	2
MOV direct,Rn	Move register to direct byte	88-8F	2	2
MOV direct1,direct2	Move direct byte to direct byte	85	3	3
MOV direct,@Ri	Move indirect RAM to direct byte	86-87	2	3
MOV direct,#data	Move immediate data to direct byte	75	3	3
MOV @Ri,A	Move accumulator to indirect RAM	F6-F7	1	2
MOV @Ri,direct	Move direct byte to indirect RAM	A6-A7	2	2
MOV @Ri,#data	Move immediate data to indirect RAM	76-77	2	2
MOV DPTR,#data16	Load data pointer with a 16-bit constant	90	3	3
MOVC A,@A+DPTR	Move code byte relative to DPTR to accumulator	93	1	4
MOVC A,@A+PC	Move code byte relative to PC to accumulator	83	1	4
MOVX A,@Ri	Move external RAM (8-bit address) to A	E2-E3	1	3*
MOVX A,@DPTR	Move external RAM (16-bit address) to A	E0	1	3*
MOVX @Ri,A	Move A to external RAM (8-bit address)	F2-F3	1	3*
MOVX @DPTR,A	Move A to external RAM (16-bit address)	F0	1	3*
PUSH direct	Push direct byte onto stack	C0	2	2
POP direct	Pop direct byte from stack	D0	2	2
XCH A,Rn	Exchange register with accumulator	C8-CF	1	2
XCH A,direct	Exchange direct byte with accumulator	C5	2	2
XCH A,@Ri	Exchange indirect RAM with accumulator	C6-C7	1	2

XCHD A,@Ri	Exchange low-order nibble indirect RAM with A	D6-D7	1	2
------------	---	-------	---	---

Table 6. Data transfer instructions

* MOVX cycles depends on STRETCH register content.

Mnemonic	Description	Code	Bytes	Clk periods
ACALL addr11	Absolute subroutine call	11-F1	2	3
LCALL addr16	Long subroutine call	03	3	4
RET	Return from subroutine	22	1	3
RETI	Return from interrupt	32	1	3
AJMP addr11	Absolute jump	01-E1	2	3
LJMP addr16	Long jump	02	3	4
SJMP rel	Short jump (relative address)	80	2	3
JMP @A+DPTR	Jump indirect relative to the DPTR	73	1	3
JZ rel	Jump if accumulator is zero	60	2	3
JNZ rel	Jump if accumulator is not zero	70	2	3
JC rel	Jump if carry flag is set	40	2	3
JNC	Jump if carry flag is not set	50	2	3
JB bit,rel	Jump if direct bit is set	20	3	4
JNB bit,rel	Jump if direct bit is not set	30	3	4
JBC bit,direct rel	Jump if direct bit is set and clear bit	10	3	4
CJNE A,direct rel	Compare direct byte to A and jump if not equal	B5	3	4
CJNE A,#data rel	Compare immediate to A and jump if not equal	B4	3	4
CJNE Rn,#data rel	Compare immediate to reg. and jump if not equal	B8-BF	3	4
CJNE @Ri,#data rel	Compare immediate to ind. and jump if not equal	B6-B7	3	4
DJNZ Rn,rel	Decrement register and jump if not zero	D8-DF	2	4
DJNZ direct,rel	Decrement direct byte and jump if not zero	D5	3	4
NOP	No operation	00	1	2

Table 7. Program jump instructions

Mnemonic	Description	Code	Bytes	Clk periods
CLR C	Clear carry flag	C3	1	2
CLR bit	Clear direct bit	C2	2	3
SETB C	Set carry flag	D3	1	2
SETB bit	Set direct bit	D2	2	3
CPL C	Complement carry flag	B3	1	2
CPL bit	Complement direct bit	B2	2	3
ANL C,bit	AND direct bit to carry flag	82	2	3
ANL C,/bit	AND complement of direct bit to carry	B0	2	3
ORL C,bit	OR direct bit to carry flag	72	2	3
ORL C,/bit	OR complement of direct bit to carry	A0	2	3
MOV C,bit	Move direct bit to carry flag	A2	2	3
MOV bit,C	Move carry flag to direct bit	92	2	3

Table 8. Boolean manipulation instructions