

Glitch Free Clock Switching Techniques in Modern Microcontrollers

Borisav Jovanović, Milunka Damnjanović

Abstract - Multi-frequency clock signals are being widely used in chips, especially in the communications area. These clock frequencies can be totally unrelated or they may be multiples of each other. In either case, there is a chance of generating a glitch on the clock line at the time when switch changes. The paper presents the implementation of glitch free clock switching techniques applied in the design of 8051 microcontroller.

Keywords - Multi-frequency clock signals, clock switching, microcontroller

I. INTRODUCTION

Multi-frequency clock signals are being widely used in digital circuits, especially in the communications area. The frequencies of these clock signals can be multiples of each other or totally unrelated. Modern microprocessors also utilize multiple clock signals. They are able to work under different load conditions, and operate at several clock frequencies. When large amount of data processing is required from microprocessor, its speed is set to the maximum level. In some other operating conditions, for example, when microprocessors are embedded in wireless sensor nodes, the power consumption is extremely reduced. The clock frequency impacts dynamic component of power dissipation and the decrease of clock frequency leads to the reduction of total power consumption.

In multi-clock signal digital systems there is a chance of generating a glitch or chopped signal on the clock line at the time when clock signal is changed. Special attention has to be paid to avoid the timing problems. The paper presents the implementation of glitch free clock switching techniques applied in the design of a microcontroller.

II. GLITCH-FREE CLOCK SWITCH CIRCUIT

The simplest clock switch is the multiplexer circuit (Fig.1). The multiplexer is comprised of AND, OR and INVERTER logical gates. The switch takes two clock signal sources at inputs (signals CLKA and CLKB). When the signal SEL value changes, the multiplexer alters the clock source input to the output. The frequencies of clock signals CLKA and CLKB can be multiples of each other, or they may be not related in any way. The select

Borisav Jovanović and Milunka Damnjanović are with the Department of Electronics, Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia, E-mail: {borisav.jovanovic, milunka.damnjanovic}@elfak.ni.ac.rs.

control signal SEL is usually generated by some sequential circuit, driven by either CLKA or CLKB.

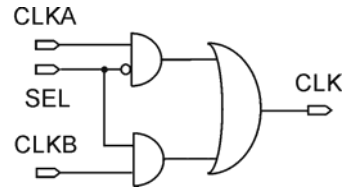


Fig. 1. The clock switch based on multiplexer circuit

Unfortunately, the switch may generate chopped clock signal or a glitch at the output CLK (Fig.2). The clock switching which occurs during output clock's high state has to be avoided, regardless of the frequency values or phase relationship of input clock signals.

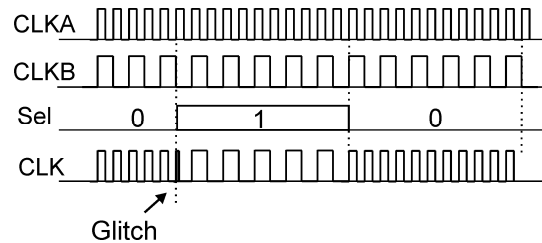


Fig. 2. The signal waveforms of multiplexer circuit

A clock switch circuit that prevents glitch generation at the output is presented in Fig. 3. [1] The circuit can be used when frequencies of input clock signals are multiples of each other. The input clock signals can be generated by some clock divider circuit.

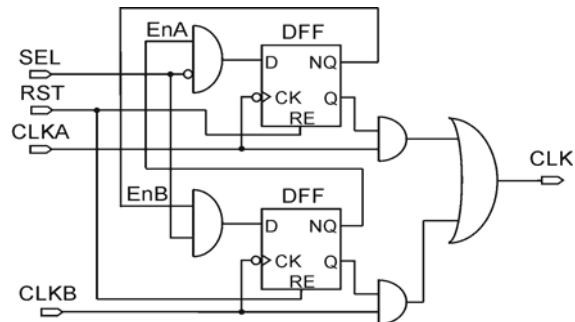


Fig. 3. Glitch-free clock switch circuit

Negative edge triggered D flip-flops are inserted in the selection path for each of the clock sources (Fig.3). D flip-flops are used to store the selection signals EnA and EnB. New values EnA and EnB are stored on negative edges of clock signals CLKB and CLKA respectively. This protects against glitches at the output CLK.

The next clock source is selected only after previous clock is deselected. Actually, the switch has to wait for deselection of the current clock before starting the propagation of the next clock source. This guarantees that no changes occur at the output while either of the clocks is at high level, thus avoiding the chopped signal at output.

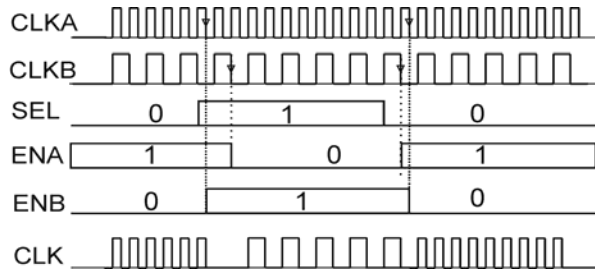


Fig. 4. The signal waveforms for the signals of glitch free-clock switch circuit

The Fig.4 presents the waveforms of clock switch circuit signals. The transition of the signal SEL from 0 to 1 first stops the propagation of CLKA to the output CLK. This happens at the preceding falling edge of CLKA. At following negative edge of CLKB, the signal ENA is reset and the propagation of CLKB is started. Now there is not any glitch or chopped signal at the clock switch output CLK.

III. THE IMPLEMENTATION OF GLITCH-FREE CLOCK SWITCH IN 8051 MICROCONTROLLER

A. The microcontroller

The microcontroller block (MCU) has standard 8051 instruction set that contains 255 different instructions. The 8051 complex instruction set is popular and widely supported by many software development tools [2, 3].

The microcontroller core is quite fast. The one-byte instructions are executed in only two clock cycles. For comparison, the execution period of one-byte instructions in industrial-standard 8051 microcontroller is 12 clock cycles.

The main structure of proposed microcontroller block consists of core, memory blocks, the block for programming and initialization and peripheral units.

The MCU core performs fetching, decoding and executing of instructions and consists of the control logic block, arithmetical-logical unit (ALU) and Special Function Register (SFR) control logic.

The peripherals are comprised of three digital

input/output parallel ports (P0, P1 and P2), several communication modules - one asynchronous universal receiver/transmitter (USART) and I2C interface. Also, three standard timer/counter circuits are present.

The memory organization is similar to that of the industry standard 8051 microcontroller. The main memory areas are:

- program memory (on-chip 8kB SRAM block),
- external data memory XRAM (physically consisting of on-chip 2kB SRAM block),
- internal data memory IRAM (comprising of on-chip 256 Internal Dual port RAM and Special Function Register block).

All SRAM memory blocks are physically located on the chip. The implemented MCU does not have non-volatile memory for program code storage. Instead, the MCU utilizes on-chip SRAM memory and an external serial 24LC64 EEPROM chip. Every time, after the reset state, the program memory is automatically loaded from external EEPROM chip into the 8kB SRAM program memory.

The microcontroller was implemented using commercial 65nm digital standard cell library and Cadence tool suite [4].

The power consumption of microcontroller can be divided into two power components: dynamic and static consumption.

The static power is caused by presence of leakage currents in MOS transistors. In new technologies the amount of leakage is rapidly increased compared to dynamic current component [5]. New technologies proposed techniques that solve the leakage problem. The design layout is divided into different power domains with separated power and ground lines [6]. The power domains may operate at different voltage supply values. Also, special transistors are inserted into domains to switch off the parts of the chip which are currently inactive [7].

Dynamic power consumption depends on operating frequency and voltage supply level. Since the implemented microcontroller works at fixed voltage supply level of 1.2V, the dynamic power consumption is reduced by utilization of clock gating techniques and decreasing the operating frequency. The proposed MCU is able to work under different load conditions, and several operating frequencies are at disposal. When large amount of processing is required, the speed is set to the maximum level of 60MHz. When processing load is low, the lower clock frequencies are used.

B. The glitch free clock divider circuit

To reduce the dynamic power in applications that do not require intensive data processing, the MCU operating frequency is reduced. The MCU incorporates clock divider circuit with six output signals (Fig. 5). The following clock signals are used: 60MHz, 30MHz, 15MHz, 7.5MHz, 3.75MHz and 1.875MHz.

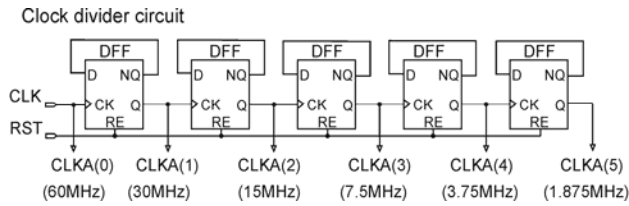


Fig. 5. The clock divider circuit implemented within the microcontroller

The operating frequency is changed instantly during program code execution. The operating speed is simply programmed by writing the new value into PMSR register. The PMSR register is one of the registers in Special Function Register set and has hexadecimal address 0x8E. The three least-significant bits of PMSR register select the frequency of MCU clock signal. The Table I gives the relation between the PMSR content and selected frequency.

TABLE I
PMSR REGISTER CONTENT FOR SELECTING THE CLOCK FREQUENCY VALUE

PMSR(2:0)	CLKA(i)
"000"	60MHz
"001"	30MHz
"010"	15MHz
"011"	7.5MHz
"100"	3.75MHz
"101"	1.875MHz

A glitch on the clock line is hazardous to the whole system, as it could be interpreted as a capture clock edge by some registers while missed by others. The circuit implementing the glitch free clock signal selection is given in Fig. 6.

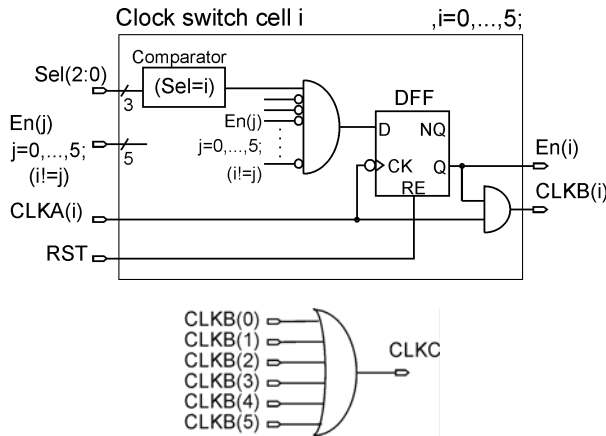


Fig. 6. The circuit for glitch-free clock signal selection implemented in microcontroller

The circuit takes at inputs the clock divider output signals CLKA(i) (i=0,...,5), and produces glitch free clock signal CLKC. The signal CLKC is fed to microcontroller clock input. The input Sel(2:0) represents the 3-bit content

of PMSR register, selecting one of the clock frequencies.

The circuit consists of six identical clock switch cells (i=0,...,5) implementing the selection path for six clock sources. Each cell enables the propagation of one clock source CLKA(i) to the switch output. The cell produces internal CLKB(i) which is fed further to the OR gate input.

The cell contains a D flip-flop which stores the clock enable signals En(i). The value En(i) is changed at negative edges of the clock signal CLKA(i). The signal En(i) is set when input signal Sel(2:0) is equal to number i and the corresponding enable signals of other cells En(j), j=0,...,5 are in the reset state.

The Fig.7 shows the glitch free clock signal CLKC, when operating speed is changed by writing different values into PMSR register.

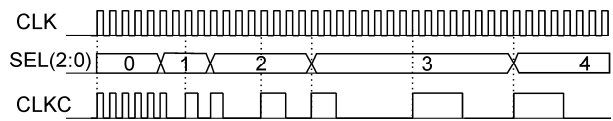


Fig. 7. The signal waveforms of glitch free-clock switch circuit implemented within the microcontroller

C. The clock gating

Clock tree power dissipation is significant component of MCU's power consumption because the clock is fed to all sequential standard cells, and the clock signal switches every cycle. Clock gating is an efficient technique for dynamic power reduction and it was often used during the microcontroller design.

To avoid glitches on clock gate output signals special cells were used. The standard cell library TCBN65LP [4], in which the microcontroller is implemented, offers two different standard cells which are used as gated clock latches:

- CKLHQ - Negative-edge gated clock latch with Q output only
- CKLNQ - Positive-edge gated clock latch with Q output Only

Both types of cells have several versions, with different drive strengths. For example, the cell CKLHQ has following variants: CKLHQD1, CKLHQD2, CKLHQD4, CKLHQD6BWP7T, CKLHQD8 (with the strongest load drive).

The clock gating cells are included in RTL descriptions of microcontroller parts by simply making instances of these cells in VHDL code:

```
Cell_lab: CKLHQ port map (TE=>sig1, E=>sig2,
CPN=>CLK, Q=>sig3);
```

Fig. 8. The specification of clock gating cell in VHDL code

The synthesis tool recognized the clock gating cells in VHDL descriptions and changed the cells with cells that have appropriate drive strengths. Unfortunately, the

standard cell library doesn't offer the clock gating latches which have the reset input. In some MCU's components, these cells were needed, so, custom made gating cells were used, which schematic is given in Fig. 9.

The clock tree was synthesized by CTS (Clock Tree Synthesis) SoC Encounter tool [8]. The clock tree generation process was directed by information written in timing constraint files. The constraints included maximum delays and skews on clock signals. The CTS tool automatically found the number of clock tree levels and balanced the clock phase delays with appropriately sized clock buffers. Besides, CTS performed timing analysis and optimization through clock gating logic.

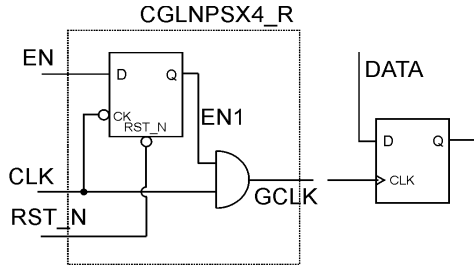


Fig.9. Custom made clock gate made of standard cells

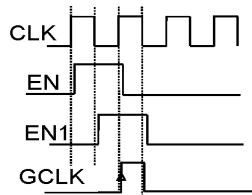


Fig.10. The signal waveforms for the signals of the clock gate circuit

After the layout was generated, the logical verification of final netlist was performed. The exact delays on signals have been extracted from layout and used together with standard cells timing information provided by timing libraries. The timing problems (setup-hold warnings and the logical errors) concerned with clock tree generation and clock gates were not present in simulations.

IV. CONCLUSION

The implemented microcontroller operates at several clock frequencies and is able to work under different load conditions.

The architecture of the microcontroller clock switch block is considered. The proposed circuit implements glitch free clock signal selection for safe MCU operation. The MCU provides six clock frequencies: 60MHz, 30MHz, 15MHz, 7.5MHz, 3.75MHz and 1.875MHz. The operating speed is simply programmed by writing the special function register.

Clock gating was often used during the microcontroller design. To avoid glitches on clock gate output signals special cells were used. The logical verification of final layout netlists proved the absence of timing problems.

ACKNOWLEDGEMENT

Results presented in this paper are part of achievements obtained within the project TR32004 funded by the Serbian Ministry of Science and Technology Development.

REFERENCES

- [1] Mahmud R., "Techniques to make clock switching glitch free ", EETimes, June 30, 2003.
- [2] KEIL C compiler and development tool for C51 microcontrollers, <http://www.keil.com/c51/devproc.asp>
- [3] SDCC 8051 compiler documentation, <http://sdcc.sourceforge.net/>
- [4] TSMC 65nm LP Standard Cell Libraries - TCBN65LP http://www.europractice-ic.com/libraries_TSMC.php
- [5] Bipul, P., Agarwal, A., Roy, K., "Low-Power Design Techniques for Scaled Technologies", Integration, The VLSI Journal 2006, pp.64–89
- [6] Katkoori, S., Roy, S., Ranganathan, N. "A Framework for Power-Gating Functional Units in Embedded Microprocessors", IEEE Transactions on VLSI Systems, 2009, Vol.17, N.11, pp.1640-164
- [7] Keating, M., Flynn, D., Aitken, R., Gibbons, A., Shi, K., "Low Power Methodology Manual", Springer, 2007
- [8] SoC Encounter User Guide - SocUG.pdf, Cadence documentation