Oregano Systems
A Meinberg Company

MC8051 IP Core

Synthesizeable VHDL Microcontroller IP-Core

User Guide

Web: https://www.oreganosystems.at/products/ip-cores/8051-ip-core
Contact: contact@oreganosystems.at

Version 1.4
March 2019

8051 IP Core - Overview

Key Features

- Fully synchronous design
- Instruction set compatible to the industry standard 8051 microcontroller
- Optimized architecture enables fast one to four clocks per OP code
- Up to 10 times faster due to completely new architecture
- User selectable number of timers/counters as well as serial interface units
- Active timer/counter and serial interface units selectable via additional special function register
- Optional implementation of the multiply command (MUL) using a parallel multiplier unit
- Optional implementation of the divide command (DIV) using a parallel divider unit
- Optional implementation of the decimal adjustment command (DA)
- No multiplexed I/O ports
- 256 bytes internal RAM
- Up to 64 Kbytes ROM and up to 64 Kbytes RAM
- Source code available free of charge under the GNU LGPL license
- Technology independent, clear structured, well commented VHDL source code
- Easily expandable by adapting/changing VHDL source code
- Parameterizeable via VHDL constants

8051 IP Core – Block Diagram

The starting from the top level module and its submodules are depicted in figure 1. The toplevel signal names are shown as well as the three memory blocks used in the design. The user selectable number of serial interfaces and timer/counter units is indicated by the dotted line between the modules *mc8051_siu* and *mc8051_tmrctr*.

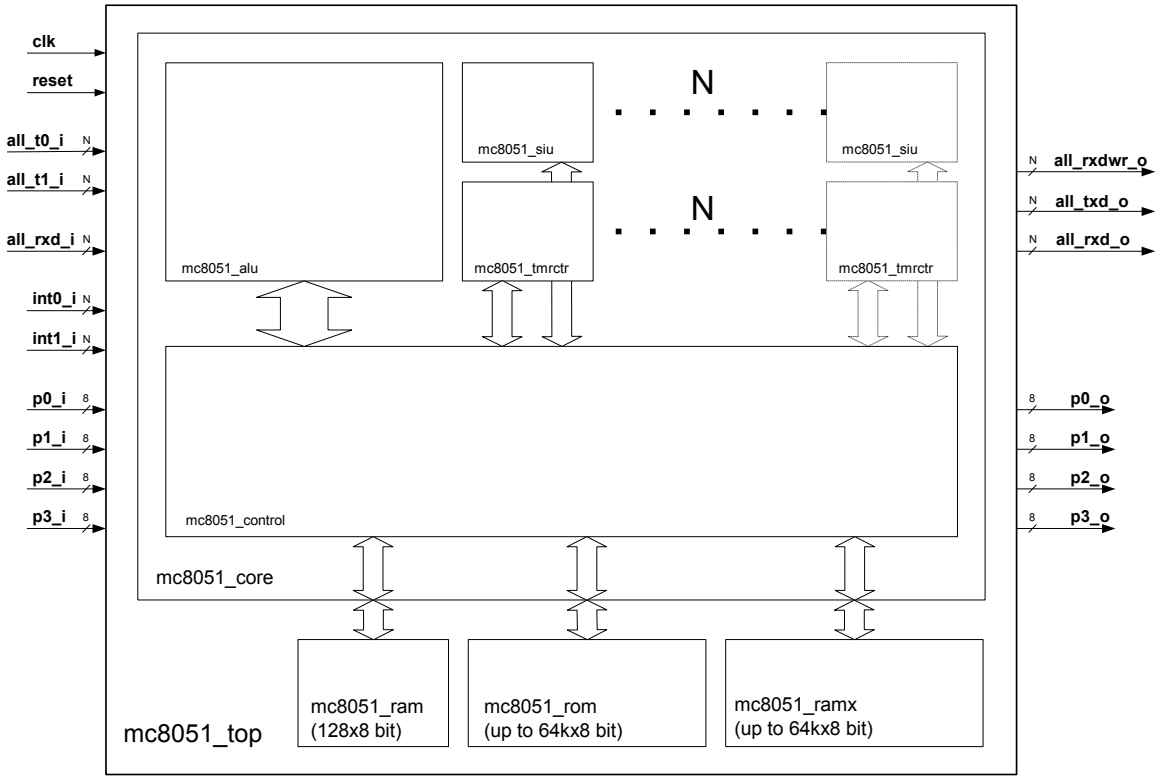

figure 1: Block diagram of the 8051 microcontroller IP-core.

Signal Name	Description
clk	System clock. Only rising edge used.
reset	Asynchronous reset of all flip-flops.
all_t0_i	Timer/counter 0 inputs.
all_t1_i	Timer/counter 1 inputs.
all_rxd_i	Receive data input for serial interface units.
int0_i	Interrupt 0 inputs.
int1_i	Interrupt 1 inputs.
p0_i	Parallel port 0 input.
p1_i	Parallel port 1 input.
p2_i	Parallel port 2 input.
p3_i	Parallel port 3 input.
all_rxdwr_o	Data direction signal for bidirectional rxd input/output (high = output) data.
all_txd_o	Transmit data output for serial interface unit.
all_rxd_o	Data output for mode 0 operation of serial interface unit.
p0_o	Parallel port 0 output.
p1_o	Parallel port 1 output.
p2_o	Parallel port 2 output.
p3_o	Parallel port 3 output.

table 1: Top level signal name.

Design Hierarchy

The design hierarchy and the corresponding VHDL files are depicted in figure 2.

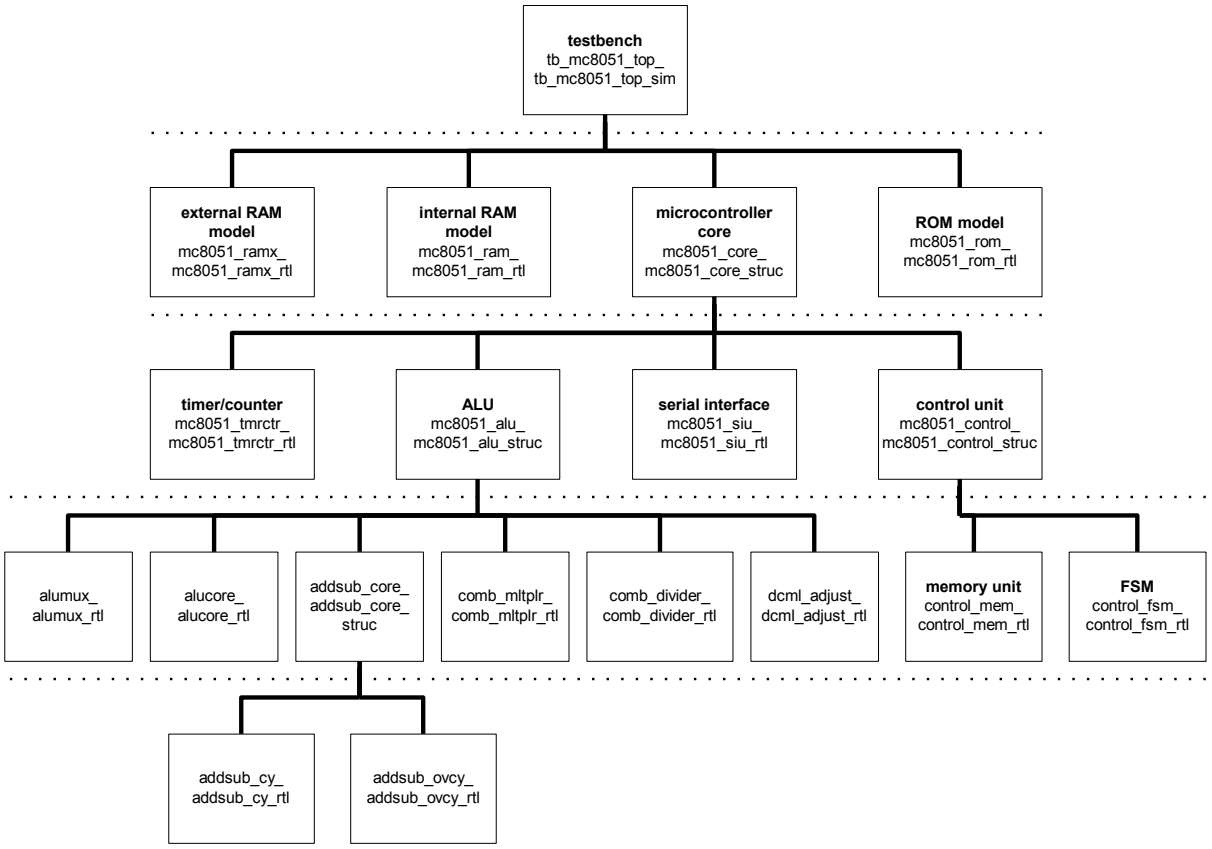

figure 2: Design hierarchy of the 8051 microcontroller IP-core.

The VHDL source files have been consistently named throughout the whole design:

- VHDL entities — *entity-name*_.vhd
- VHDL architectures — *entity-name*_rtl.vhd for modules containing logic
- *entity-name*_struc.vhd for modules just connecting submodules
- VHDL configurations — *entity-name*_rtl_cfg.vhd
  *entity-name*_struc_cfg.vhd

The core itself is made up of the submodules timer/counter, ALU, serial interface, and control unit. RAM or ROM blocks are most often generated corresponding to the selected target technology and are therefore instantiated in the highest design hierarchy. Generated RAM blocks and BIST structures - for ASIC production test integration - can be easily added at this level of the design.

Clock Domains

The 8051 IP core is a fully synchronous design. There is a single clock signal that controls the clock input of every storage element. Clock gating is not used. The clock signal is not fed into any combinatorial element. The interrupt input lines are synchronized to the global clock signal using a standard two-level synchronization stage because they may be driven by external circuitry that operates with another clock. The parallel port input signals are not synchronized that way. If the user decides that there is also the need for synchronizing these signals it may be added easily.

Memory Interfaces

Due to the optimized architecture the signals coming from and going to the memory blocks have not been registered. So during synthesis input and output timing constraints should be placed on the corresponding ports and synchronous memory blocks should be used for the mc8051 IP-core.

Configuring the 8051 IP Core

In the following the parameterizability of the 8051 microcontroller IP-core design will be discussed and information for embedding the IP-core in larger designs will be given.

Timer/Counter, Serial Interface, and Interrupts

The original microcontroller design offered only 2 timer/counter units, one serial interface, and two external interrupt sources. 8051 derivates later offered more of these resources on chip. Since this is sometimes a limiting factor we decided to implement some sort of parameterization in the 8051 IP core. This 8051 microcontroller IP-core offers the capability to generate up to 256 of these units by simply changing a VHDL constant's value.

In the VHDL source file mc8051_p.vhd the constant C_IMPL_N_TMR can take values from 1 to 256 to control this feature. Values out of this interval result in a non functioning configuration of the core. Figure 3 shows the corresponding lines of VHDL code.

```
------------------------------------------------------------------------------
-- Select how many timer/counter units should be implemented
-- Default: 1
constant C_IMPL_N_TMR : integer := 1;
------------------------------------------------------------------------------

------------------------------------------------------------------------------
-- Select how many serial interface units should be implemented
-- Default: C_IMPL_N_TMR ---(DO NOT CHANGE!)---
constant C_IMPL_N_SIU : integer := C_IMPL_N_TMR;
------------------------------------------------------------------------------

------------------------------------------------------------------------------
-- Select how many external interrupt-inputs should be implemented
-- Default: C_IMPL_N_TMR ---(DO NOT CHANGE!)---
constant C_IMPL_N_EXT : integer := C_IMPL_N_TMR;
------------------------------------------------------------------------------
```

figure 3: VHDL source code for configuring the number of timer/counter units, serial interfaces, and external interrupts.

At the moment the three constants `C_IMPL_N_TMR`, `C_IMPL_N_SIU`, `C_IMPL_N_EXT` cannot be changed independently. Incrementing constant `C_IMPL_N_TMR` by one means to generate two additional timer/counter units, one additional serial interface, and two additional external interrupt sources.

To be able to reach all registers of the generated units without changing the address space of the microcontroller only two 8bit registers are inferred as additional special function registers. These are TSEL (address 0x8Eh for timer/counter units) and SSEL (address 0x9Ah for serial interface units). If these registers point to a not existent device number, the default unit number 1 is selected. The circuit is depicted in figure 4.

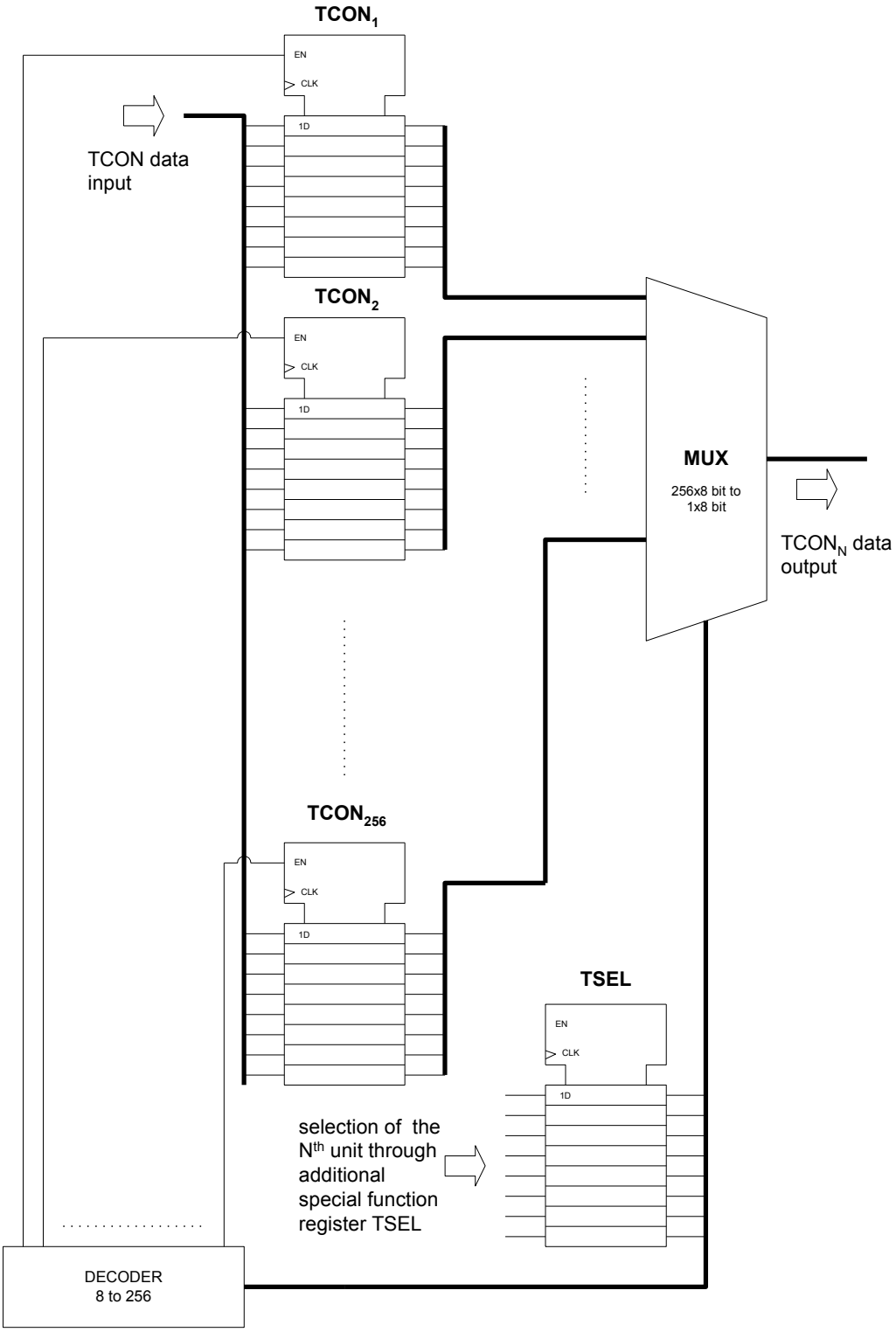

figure 4: Selection of a TCON register using additional TSEL register.

If an interrupt occurred during this very device was not selected by e.g. TSEL, the corresponding interrupt flag stays set until the matching interrupt service routine is executed. Subsequent interrupts during the time the device has not been selected thou result in only one single call to the interrupt service routine.

Optional Instructions

In some cases, it makes sense to not implement instructions which are not needed and consume furthermore much chip area. Such instructions are 8bit multiplication, 8bit division, and 8bit decimal correction. Therefore, the MUL instruction for 8bit multiplication can be skipped when the VHDL constant `C_IMPL_MUL` in the mc8051_p.vhd source file is set to 0. Equally the 8bit division DIV can be skipped through setting the VHDL constant `C_IMPL_DIV` to 0 and the decimal correction instruction can be skipped by setting the constant `C_IMPL_DA` to 0. The corresponding lines of VHDL source code can be seen in figure 5.

```
------------------------------------------------------------------------------
-- Select whether to implement (1) or skip (0) the multiplier
-- Default: 1
constant C_IMPL_MUL : integer := 1;
------------------------------------------------------------------------------

------------------------------------------------------------------------------
-- Select whether to implement (1) or skip (0) the divider
-- Default: 1
constant C_IMPL_DIV : integer := 1;
------------------------------------------------------------------------------

------------------------------------------------------------------------------
-- Select whether to implement (1) or skip (0) the decimal adjustment command
-- Default: 1
constant C_IMPL_DA  : integer := 1;
------------------------------------------------------------------------------
```

figure 5: Code fragment showing how instructions can be skipped.

The gain in terms of chip area when not implementing all three optional instructions is approximately 10 %.

Parallel I/O Ports

The mc8051 IP-core offers just as the original 8051 microcontroller 4 bidirectional 8bit I/O ports to conveniently exchange data with the microcontroller's environment. To ease integration of our core for IC design the original's multi-function ports have not been rebuilt and all signals (e.g. serial interface, interrupts, counter inputs, and interface to external memory) have been fed separately out of the core (see figure 1). The basic structure of the parallel I/O ports is shown in figure 6.

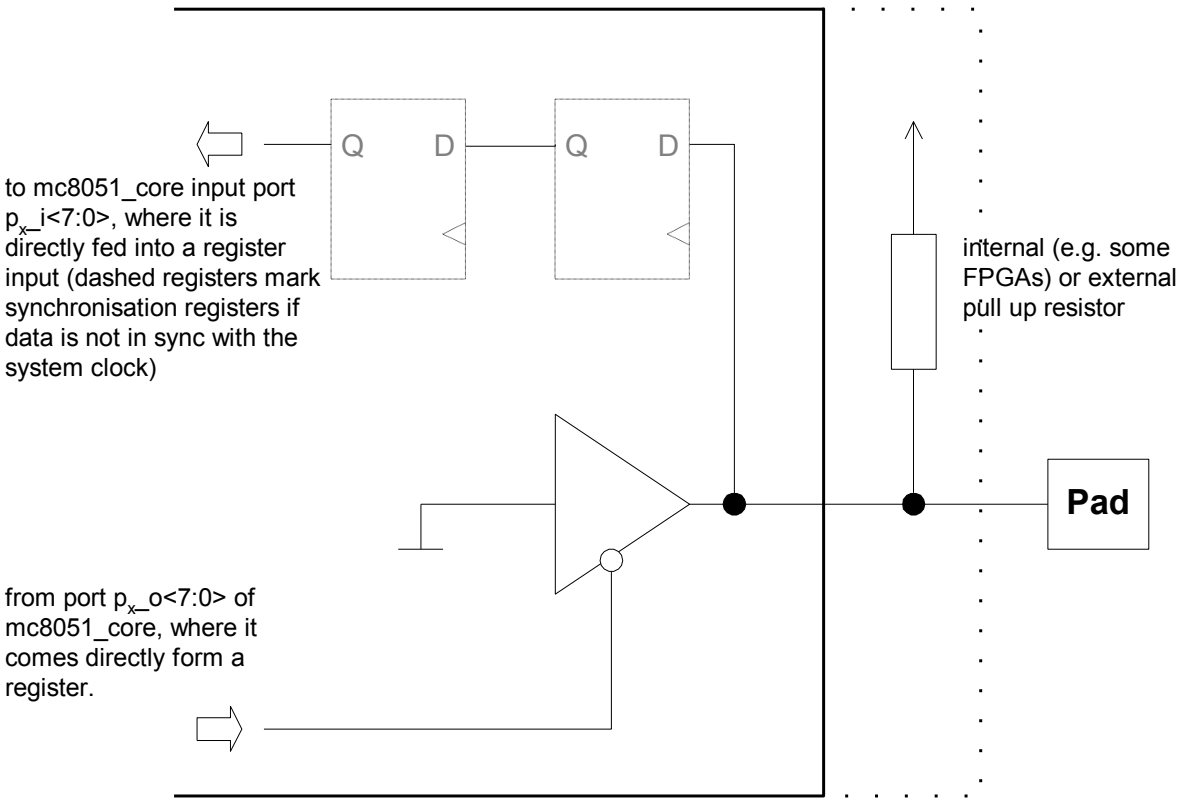

figure 6: Basic structure of the parallel I/O ports.

Verification

Verification of the core was accomplished by simulating the VHDL code and comparing the results of the executed program (i.e. ROM contents) with the results produced by an industry standard 8051 simulator (http://www.keil.com/demo/evaldl.asp?p=C51).

After simulation the contents of a certain memory area is written to a file both for the standard 8051 simulator (using the command `save keil.hex 0x00,0xFF`) and the VHDL code simulation (e.g. using the script `write2file.do`, producing the file `regs.log` in the simulation directory). The resulting text files have to be identical.

To be able to feed the compiled assembler file from the Keil development software to the VHDL code simulation a short C program is provided in the latest distribution of the IP core. It converts the Intel hex format file into a text file containing binary 8 bit data, suited for being read by the VHDL simulator (file `mc8051_rom.dua` in the simulation directory).

Deliverables

Extract the mc8051.zip file as is using the directory tree we recommend. There are already scripts for synthesis using Synopsys's DesignCompiler for ASIC design and Synplicity's Synplify for FPGA design. Additionally there are scripts for RTL simulation using Mentor's/Modeltech's Modelsim. See figure 7 for an overview of the mc8051 directory tree.

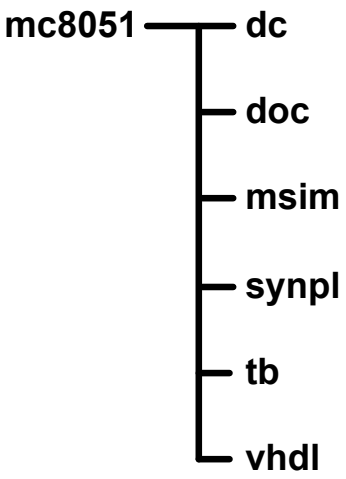

figure 7: Directory tree in which the mc8051 IP-core is distributed.

Directory	Contents
dc	DC synthesis script files
doc	Documentation
msim	Modelsim simulation directory
synpl	Synthesis with Synplicity's Synplify
tb	Testbench and memory models
vhdl	Synthesizeable, well commented VHDL source code

table 2: Directory tree: Brief description.

Document Revisions

- Version 1.0, January 2002: Initial version of the mc8051 IP Core User Guide.
- Version 1.1, June 2002: Added document history and added some details to the verification paragraph. Removed `scr` directory from project tree.
- Version 1.2, June 2013: Changed mail address, Web link and project tree