



OpenCores.Org

T48 μ Controller Integration Manual

Author: Arnim Luger
arniml@opencores.org

Rev 1.2
December 29, 2022

This page has been intentionally left blank.

Revision History

Rev.	Date	Author	Description
0.1	19-Jun-2005	A. Luger	First Draft
0.2	12-Sep-2005	A. Luger	Added design hierarchy, memory integration, I/O interfaces and sample systems.
0.3	31-Oct-2005	A. Luger	Description of Wishbone Master, added index.
0.4	05-Jul-2006	A. Luger	Clocking concept revised. Architectural overview added. Description of generic parameter list.
1.0	17-Dec-2006	A. Luger	T8243 added.
1.1	01-May-2008	A. Luger	Hierarchy update, RAM and ROM clarification.
1.2	29-Dec-2022	A. Luger	Add support for UPI.

Contents

INTRODUCTION.....	1
ARCHITECTURE.....	2
CLOCKS.....	4
PORT LIST.....	6
GENERIC PARAMETERS.....	9
MEMORY INTEGRATION.....	11
I/O INTERFACES.....	12
T8243 I/O EXPANDER.....	13

1

Introduction

The T48 μ Controller core is an implementation of the MCS-48 and UPI-41/UPI-42 micro controller family architecture. While being a controller core for SoC, it also aims for code-compatibility and cycle-accuracy so that it can be used as a drop-in replacement for any MCS-48 controller.

The core can be configured to better suit the requirements and characteristics of the integrating system. On the other hand, nearly the full functionality of a stock 8048/8049 is available. This flexibility is achieved by separating system aspects from the core's functionality. Among others, this includes memory sizes, memory implementation and clock generation.

For reference and to enable quick setup, this core is accompanied by several sample systems. They demonstrate how the configuration features can be utilized to tailor the core to one's needs.

The T48 μ Controller project is maintained at

<https://opencores.org/projects/t48>

Updates of the core can be obtained via the project pages.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

2

Architecture

Based on the original MCS-48 architecture specification, the T48 μ Controller includes all modules of this family as depicted in the following 1. For functional details refer to the “MCS-48 Microcomputer User's Manual”.

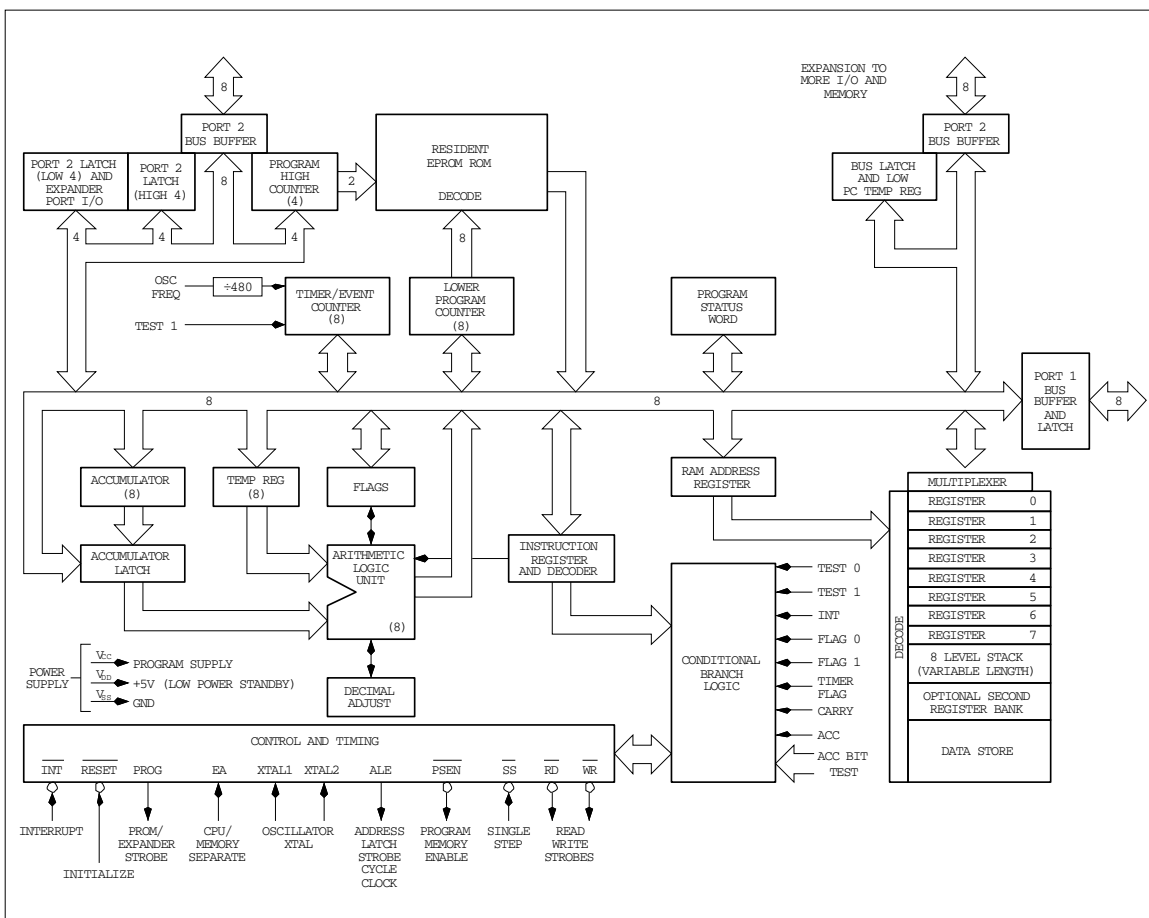


Figure 1: Block Diagram

This architectural structure has been partitioned into sub-modules as shown in 2.

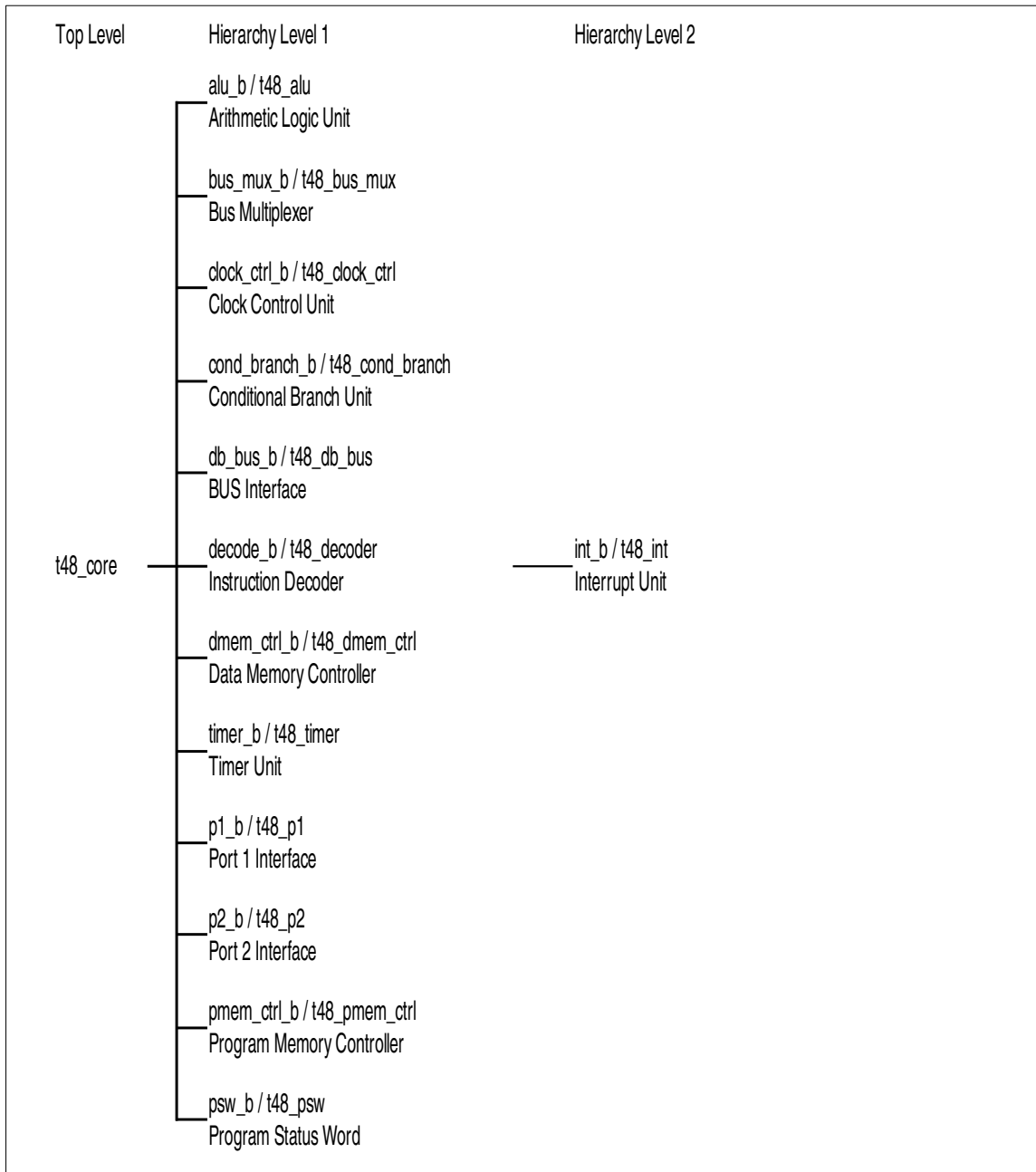


Figure 2: T48 μ Controller Hierarchy

The upi41_core architecture replaces t48_db_bus with upi41_db_bus.

3

Clocks

The T48 μ Controller core operates on one single clock. However, due to the characteristics of the MCS-48 clocking system, this main clock is derived by a clock divider. This section explains the details.

Name	Source	Rates (MHz)		Description
		Max	Min	
xtal_i	Input Pad	$n \times f_{\max}$	-	Clock input from external crystal / clock generation circuit.
clk_i	Clock former	f_{xtal_i}	$f_{\text{xtal}_i}/3$	Main system clock. Synchronously enabled by en_clk_i.

Table 1: List of clocks

The main clock is applied at input xtal_i. To support system integration, this main clock can be synchronously enabled/disabled with the xtal_en_i input. A '1' on this pin qualifies the next rising edge on xtal_i as a valid edge for the whole core logic. In contrast, a '0' will cause the core to ignore this edge and halt operation until xtal_en_i is '1'.

The clocking system of the MCS-48 family establishes a circuit that divides the incoming external clock on XTAL by 3 to generate the base clock for all system operations. The T48 μ Controller core needs to mimic this scheme because most of the control signals like psen_n_o, rd_n_o etc. are generated with the clock on xtal_i. This is all done inside the clock_ctrl module.

All other logic of the T48 μ Controller operates with the main system clock applied to clk_i. It is the responsibility of the system to provide a suitable clock waveform at this input. The core supports this task by providing the xtal3_o output which indicates that the next rising edge of xtal_i is the third in a row. So most of the dividing is already prepared inside the core (namely clock_ctrl). What is left to the integrating system is the final clock shaping.

There are two methods to generate the required clock at clk_i:

1. Use clock enable input en_clk_i

All modules that operate on clk_i also use en_clk_i as a synchronous clock enable. It is therefore possible to apply the external clock (connected to xtal_i) to clk_i as well, while the divider output xtal3_o is connected to en_clk_i.

This scenario is the more simple one and should work with any FPGA technology.

2. Shape external XTAL clock by clock gating

In case your technology provides valid clock gating circuitry, you can gate the external XTAL, thus generating a divided clock at `clk_i`. Use `xtal3_o` as the clock gate enable signal. As `clk_i` is already supplied with the required clock, the synchronous clock enable at `en_clk_i` has to be tied constantly to '1'.

This option is the most elegant one as it will result in reduced area (synchronous clock enables optimized away from each flip-flop) and reduced power consumption (flip-flops are only clocked every third clock). However, dedicated clock gating support from the underlying technology is required to safely gate the incoming clock without glitches.

4

Port List

This section specifies the I/O ports of the T48 μ Controller.

Port	W	Dir	Description
T48 Interface			
xtal_i	1	In	Clock from external crystal/clock generation circuit.
xtal_en_i	1	In	Synchronous clock enable for xtal_en_i
reset_i	1	In	Asynchronous reset input.
t0_i	1	In	Test 0 input.
t0_o	1	Out	Test 0 output (derived clock output).
t0_dir_o	1	Out	Direction selector for T0 pad. 0 ... T0 is operated in input direction 1 ... T0 is in output mode
int_n_i	1	In	Interrupt input. (Active low)
ea_i	1	In	External Access input which forces all program memory fetches to reference external memory.
rd_n_o	1	Out	Output strobe activated during a BUS read. (Active low)
pсен_n_o	1	Out	Program Store Enable. This output occurs only during a fetch to external program memory. (Active low)
wr_n_o	1	Out	Output strobe during a BUS write. (Active low) Used as a write strobe to external data memory.
ale_o	1	Out	Address Latch Enable. This signal occurs once during each cycle. The negative edge of ALE strobes address into external data and program memory.
db_i	8	In	Data Bus or general purpose input/output bus. Read while rd_n_o is active, written while wr_n_o active. Contains the 8 low order program counter bits during an external program memory fetch, and receives the addressed instruction under the control of PSEN'. Also contains the address and data during an external RAM data store instruction, under control of ALE, RD' and WR'.
db_o	8	Out	
db_dir_o	1	Out	Direction of DB pads 0 ... DB[7..0] are operated in input direction 1 ... DB[7..0] are in output mode

Port	W	Dir	Description
t1_i	1	In	Test 1 input
p2_i	8	In	8-bit general purpose input/output port.
p2_o	8	Out	P2[3..0] contain the four high order program counter bits during an external program memory fetch and serve as a 4-bit I/O expander bus for 8243.
p2_low_imp_o	1	Out	Low impedance output driver enable for Port 2.
p1_i	8	In	8-bit general purpose input/output port.
p1_o	8	Out	
p1_low_imp_o	1	Out	Low impedance output driver enable for Port 1.
prog_n_o	1	Out	Output strobe for 8243 I/O expander.
Core Interface			
clk_i	1	In	Main core clock.
en_clk_i	1	In	Clock enable.
xtal3_o	1	Out	Indication of third XTAL clock state.
dmem_addr_o	8	Out	Data Memory address.
dmem_we_o	1	Out	Data Memory write enable.
dmem_data_i	8	In	Data Memory data input.
dmem_data_o	8	Out	Data Memory data output.
pmem_addr_o	12	Out	Program Memory address.
pmem_data_i	8	In	Program Memory data input.

Table 2: List of MCS-48 IO ports

Port	W	Dir	Description
UPI-41 Interface			
xtal_i	1	In	Clock from external crystal/clock generation circuit.
xtal_en_i	1	In	Synchronous clock enable for xtal_en_i
reset_i	1	In	Asynchronous reset input.
t0_i	1	In	Test 0 input.
cs_n_i	1	In	Chip select input. (Active low)
rd_n_i	1	In	Read strobe input. (Active low)
a0_i	1	In	Command/data select input.
wr_n_i	1	In	Write strobe input. (Active low)
sync_o	1	Out	Output clock.
db_i	8	In	Data Bus.
db_o	8	Out	Data on db_i is expected when cs_n_i and wr_n_i are active or DACK' (P2[7]) and wr_n_i are active. Data on db_o is valid when cs_n_i and rd_n_i are active or DACK' (P2[7]) and rd_n_i are active.
db_dir_o	1	Out	Direction of DB pads 0 ... DB[7..0] are operated in input direction 1 ... DB[7..0] are in output mode

Port	W	Dir	Description
t1_i	1	In	Test 1 input
p2_i	8	In	8-bit general purpose input/output port.
p2_0	8	Out	P2[3..0] contain the four high order program counter bits during an external program memory fetch and serve as a 4-bit I/O expander bus for 8243.
p2_low_imp_o	1	Out	Low impedance output driver enable for Port 2.
p1_i	8	In	8-bit general purpose input/output port.
p1_o	8	Out	
p1_low_imp_o	1	Out	Low impedance output driver enable for Port 1.
prog_n_o	1	Out	Output strobe for 8243 I/O expander.
Core Interface			
clk_i	1	In	Main core clock.
en_clk_i	1	In	Clock enable.
xtal3_o	1	Out	Indication of third XTAL clock state.
dmem_addr_o	8	Out	Data Memory address.
dmem_we_o	1	Out	Data Memory write enable.
dmem_data_i	8	In	Data Memory data input.
dmem_data_o	8	Out	Data Memory data output.
pmem_addr_o	12	Out	Program Memory address.
pmem_data_i	8	In	Program Memory data input.

Table 3: List of UPI-41 IO ports

5

Generic Parameters

This section describes the generic parameters of the T48 μ Controller.

Generic Name	Value	Description
xtal_div_3_g	1	Divide xtal_i by 3 to derive internal clock states. This setting is mandatory to maintain the original MCS-48 timing.
	0	Use xtal_i directly.
register_mnemonic_g	1	Register mnemonic output from opcode decoder, recommended.
	0	Do not register mnemonic information.
include_port1_g	1	Include Port 1 module.
	0	Do not include Port 1 module.
include_port2_g	1	Include Port 2 module.
	0	Do not include Port 2 module.
include_bus_g	1	Include BUS module.
	0	Do not include BUS module.
include_timer_g	1	Include timer module.
	0	Do not include timer module.
sample_t1_state_g	4	Sample T1 input in machine state 4. Default setting.
	3	Sample T1 input in state 3. Valid for old MCS-48 devices.

Table 4: List of MCS-48 Generic Parameters

Generic Name	Value	Description
xtal_div_3_g	1	Divide xtal_i by 3 to derive internal clock states. This setting is mandatory to maintain the original MCS-48 timing.
	0	Use xtal_i directly.
register_mnemonic_g	1	Register mnemonic output from opcode decoder, recommended.
	0	Do not register mnemonic information.

Generic Name	Value	Description
sample_t1_state_g	4	Sample T1 input in machine state 4. Default setting.
	3	Sample T1 input in state 3. Valid for old MCS-48 devices.
is_upi_type_a_g	1	Implement STS, flags and DMA features of UPI-41A and UPI-42.
	0	Do not implement STS, flags and DMA.

Table 5: List of UPI-41 Generic Parameters

6

Memory Integration

The typical configuration of the T48 μ Controller contains one ROM and one RAM module used for the Program Memory and the Data Memory, respectively. Both components have the same characteristics in that they are synchronous memories clocked by the global system clock `clk_i`. Read and write operations require a single rising clock edge, while the read-during-write characteristic does not matter.

Maximum memory sizes are constrained by the architecture of the MCS-48 family. The Data Memory can contain up to 256 bytes and the Program Memory up to 4096 bytes. The minimum size for the Data Memory is 32 bytes, whereas the members of the MCS-48 family contain at least 64 bytes. Implementation of a Program Memory component is optional. It is in the responsibility of the integrator to choose suitable memory sizes.

The T48 μ Controller's interface to the Data Memory consists of the ports `dmem_addr_o`, `dmem_we_o`, `dmem_data_i` and `dmem_data_o`. The Program Memory is interfaced via ports `pmem_addr_o` and `pmem_data_i`. Refer also to 2.

Apart from the memories' interface signals, the port `ea_i` has to be considered when integrating the Program Memory. Logic controlling `ea_i` has to implement the following scheme:

1. Port `ea_i` is set to '1' whenever the Program Memory is disabled globally. This implements the behavior of the MCS-48 EA pin.
2. Port `ea_i` is set to '1' whenever an access to a Program Memory location is announced by `pmem_address_o` that is beyond the implemented ROM size. This implements the automatic Program Memory extension of the MCS-48 family.
3. Port `ea_i` is set to '0' in all other situations.

For more details on generating `ea_i` control refer to the sample systems that come with the source code release.

7

I/O Interfaces

The MCS-48 family micro controllers contain three types of I/O interfaces.

1. Two pseudo-bidirectional general purpose I/O ports called Port 1 and Port 2.
2. One bidirectional port called BUS.
3. Two test inputs called T1 and T2.

Each of the bidirectional port is implemented as two unidirectional buses at the T48 μ Controller interface together with output enable signals. For BUS, the signal `db_dir_o` indicates, when set to '1', that all bits of BUS are operated in output mode.

The situation at Port 1 and Port 2 is a bit more complex. MCS-48 controllers implement open-drain type output drivers with pull-up resistors. This behavior can easily be built in FPGA devices with three-state drivers where the output enable control for each pin is derived from the state of the respective data bit. In addition, a high level is driven actively when the port register is written to with a '1' by the CPU. This ensures a proper transition from low to high in contrast to loading the parasitic capacitance at the pin with the pull-up resistor. To enable this behavior, dedicated control signals are available that indicate when Port 1 or Port 2 outputs should be driven actively.

3 shows a sample circuit for a bidirectional implementation of Port 1 and Port 2.

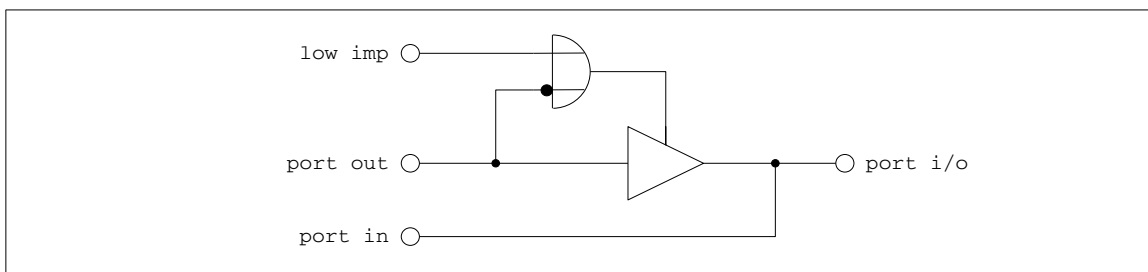


Figure 3: Pseudo-Bidirectional Port Circuit

8

T8243 I/O Expander

The T8243 core implements the functionality of the 8243 I/O expander component. Like the T48 μ Controller, it consists of a core design called `t8243_core` that is embedded in several top levels. They differ in the used clocking style: synchronous or asynchronous.

The more simple synchronous clocking used by `t8243_sync_notri` allows seamless integration in SoC designs. On the other hand, it requires an additional clock input for synchronous operation.

When choosing the asynchronous `t8243_async_notri` and `t8243` top levels, the sequential elements inside the core are clocked exclusively by the PROG input. This is closer to the original 8243 chip but imposes significant effort to obtain a robust `t8243` top level with bidirectional P2 port. Due to P2 output data being enabled as soon as PROG is asserted low, there might happen bus contention on P2 while P2 input data is being sampled by the core control logic (upon falling PROG).

6 shows the port list of the `t8243_core`.

Port	W	Dir	Description
Generic Parameters			
<code>clk_fall_level_g</code>			Active edge of flip-flops clocked by falling <code>clk_i</code> : 0 : falling edge 1 : rising edge
System Interface			
<code>clk_i</code>	1	In	Clock input.
<code>clk_rise_en_i</code>	1	In	Clock enable for rising edge triggered flip-flops.
<code>clk_fall_en_i</code>	1	In	Clock enable for falling edge triggered flip-flops.
<code>reset_n_i</code>	1	In	Asynchronous reset, low active.
Control Interface			
<code>cs_n_i</code>	1	In	Chip select.
<code>prog_n_i</code>	1	In	PROG input.
Port 2 Interface			
<code>p2_i</code>	4	In	Port 2 input bus.
<code>p2_o</code>	4	Out	Port 2 output bus.
<code>p2_en_i</code>	1	Out	Port 2 output enable.

Port	W	Dir	Description
Port 4 Interface			
p4_i	4	In	Port 4 input bus.
p4_o	4	Out	Port 4 output bus.
p4_en_i	1	Out	Port 4 output enable.
Port 5 Interface			
p5_i	4	In	Port 5 input bus.
p5_o	4	Out	Port 5 output bus.
p5_en_i	1	Out	Port 5 output enable.
Port 6 Interface			
p6_i	4	In	Port 6 input bus.
p6_o	4	Out	Port 6 output bus.
p6_en_i	1	Out	Port 6 output enable.
Port 7 Interface			
p7_i	4	In	Port 7 input bus.
p7_o	4	Out	Port 7 output bus.
p7_en_i	1	Out	Port 7 output enable.

Table 6: List of t8243_core IO ports

Appendix A

Sample Systems

Included in the release of the T48 μ Controller project, several sample systems are available. Systems building an MCS-48 compatible chip have a two-level hierarchical structure. The lower level (marked by the “notri” infix) instantiates the T48 μ Controller core and attaches the memories to the core. This level provides the unidirectional interface ports towards the system top level. Here, the interfaces are combined to bidirectional buses by three-state drivers. Chapter I/O Interfaces describes the characteristics of these drivers.

The following sample systems are available:

Name	RAM Size	ROM Size	Remark
t8039	128	None	8039HL-alike top level
t8048	64	1024	8048H-alike top level
t8050_wb	256	4096	8050AH-alike top level with Wishbone Interface

Appendix B

Wishbone Master

The Wishbone master is an optional module that can be attached to the T48 μ Controller core and enables interfacing to Wishbone compatible peripherals. Characteristics are as follows:

- Data bus 8 bit
- Address bus 24 bit
- Standard read/write cycles with wait states

The current implementation of the Wishbone master module requires exclusive access to the BUS interface of the T48 μ Controller. Refer to the t8050_wb sample system for information on how the Wishbone master module is connected to BUS. All MOVX read and write operations generate Wishbone bus cycles at the specified address. This address is built as follows:

$$\text{Wishbone address} = \text{adr2} \ \& \ \text{adr1} \ \& \ \text{address of MOVX}$$

Address components adr1 and adr2 are specified via the configuration range of the module. The following 1 summarizes the access scheme.

adr_i	MOVX Address	Description	
1	000h	Read/write adr1	Configuration Range
	001h	Read/write adr2	
0	0XXh	Wishbone cycle @ 0XXh	Wishbone Range

Table 7: Wishbone Master Access Matrix

The range selection input adr_i is controlled by P2.4.

Index

Index

8		
8039HL	15	
8048H	15	
8050AH	15	
8243	13	
B		
BUS	12, 16	
C		
Clock		
asynchronous	13	
divider	4	
gating	5	
synchronous	13	
system	4	
E		
EA	11	
G		
general purpose I/O	12	
M		
Memory		
Data	11	
Program	11	
MOVX	16	
O		
open-drain	12	
P		
peripherals	16	
Ports		
Port 1	12	
Port 2	12f.	
Port 4	14	
Port 5	14	
Port 6	14	
Port 7	14	
PROG	13	
pull-up	12	
S		
Sample System		
t8039	15	
t8048	15	
t8050_wb	15	
SoC	1	
T		
T1	12	
T2	12	
three-state	12, 15	
W		
Wishbone Master	16	
X		
XTAL	4	

