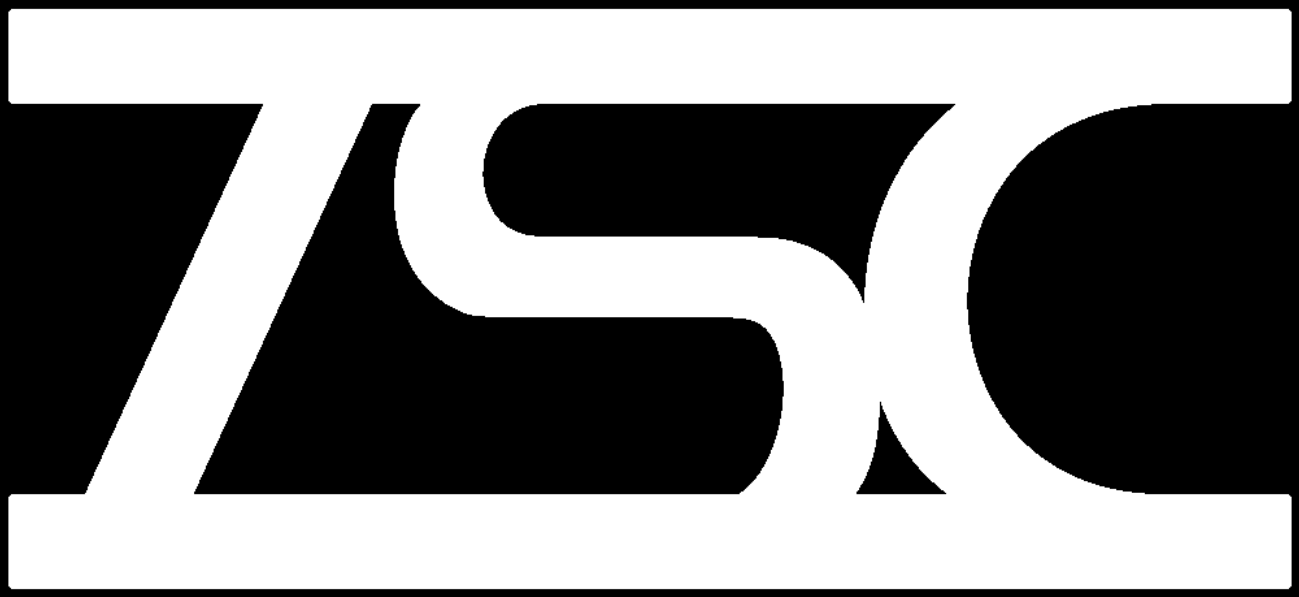


MICROCOMPUTERS

MICROCOMPUTERS

MICROCOMPUTERS

MICROCOMPUTERS

LWC31
HARDWARE & PROGRAMMING MANUAL

TURNER SEMICONDUCTOR

LWC31 Hardware & Programming Manual

Revision 3

22/ 10 / 2024

LWC31 Features

- Sixteen bit parallel processing
- 16 instructions
- 16 ALU Operations
- 13 General purpose registers
- 5 addressing modes
- True indexing capabilities
- Programmable stack pointer
- Variable length stack
- Interrupt capability
- Use with any type or speed memory
- 16 bit bi-directional Data Bus
- Addressable memory range of up to 65K words
- Turbo mode
- Bus active output

Pinout

There are 40 pins, the pins are as following (In order from left to right, looking at the interface): A0-A15 (Address bus), Write, Read, D0-D15 (Data bus), Turbo, Reset, Clock, Bus active, Set carry, Interrupt request.

Address bus (A0-A15):

16 bit address to allow up to 65K of addressable memory at one time.

Write:

While this pin is high, the LWC31 is outputting data ready to be written to the data bus.

Read:

While this pin is high, the LWC31 is reading data from the specified address.

Data bus (D0-D15):

16 bit bi-directional data bus, transferring data to and from peripherals.

Turbo:

While this pin is held high, instructions which do not use the data bus during the execution cycle will complete the execution phase on the next fetch, saving one clock cycle. On average this is a 50 percent speed increase.

Reset:

This input is used to initialize the CPU from a halt state, while this pin is held high, the CPU will not execute any instructions, the data bus is set to LOW, the address bus is set to 0xFFFF, and the read pin is set to HIGH. The clock cycle after the reset pin is LOW will set the program counter (PC) to the value on the data bus, the address bus still at 0xFFFF. The reset cycle will disable interrupts, but the registers and stack pointer are not reset.

Clock:

Clock input to the CPU

Set carry:

Rising edge on this pin will set the carry bit in the status register.

Interrupt:

While this pin is HIGH, interrupt disable bit is cleared, and the CPU is in the fetch phase, 0x0000 (BRK) is force loaded into the instruction register.

Addressing modes

Immediate addressing:

This addressing mode is available to all instructions, the operand is contained in the second word of the instruction and no further memory addressing is required.

Absolute addressing:

Absolute addressing is only available to LOD (Load) and STO (Store) instructions. In this addressing mode, data is read/written to memory.

Indexed addressing:

Index addressing is only available to LOD and STO instructions. This addressing mode is used in conjunction with registers to read/write to memory at an indexed location. The effective address is calculated by adding the base address to the contents in a register.

Register addressing:

This form of addressing is represented with a one word instruction, implying an operation with one or more registers.

Implied addressing:

In the implied addressing mode, the address containing the operand is implicitly stated in the operation code of the instruction.

Reset, fetch, execute, and interrupt cycles

Reset:

When the reset pin is pulled HIGH, the CPU will read from address 0xFFFF as the start vector to begin execution. The next clock cycle after the reset pin is LOW the CPU will latch the start vector to the PC.

Fetch:

During the fetch cycle, the CPU outputs the current program counter to the address bus to read the next opcode. When the clock signal is pulled HIGH, the CPU latches the opcode into the instruction register, the PC is incremented, and the CPU starts to read the next value as immediate data. When the clock goes back to low, the CPU latches the immediate data, the PC is incremented if immediate data is used in the opcode (otherwise the latched data is discarded), and the CPU is put into the execute phase. If the turbo pin is HIGH and the instruction does not access memory, the CPU returns immediately back to the fetch phase, and the instruction is executed during this next fetch.

Execute:

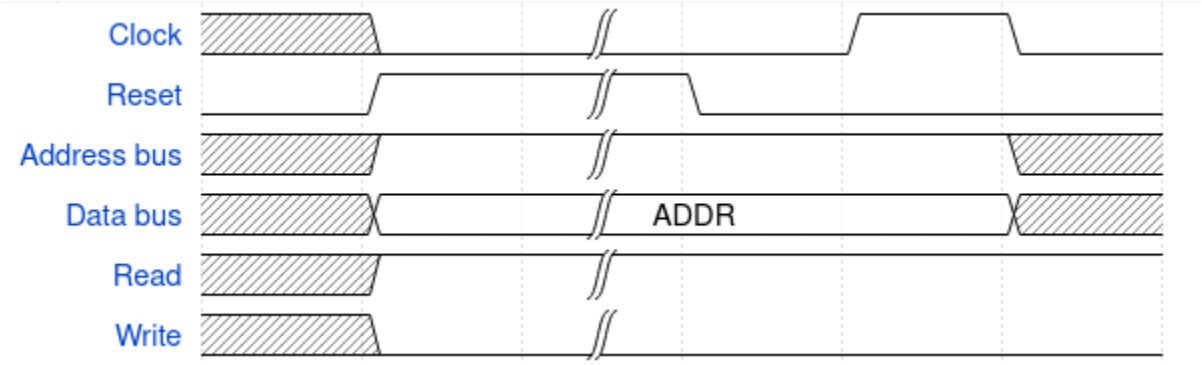
During the execute phase, the instruction is executed.

Interrupt:

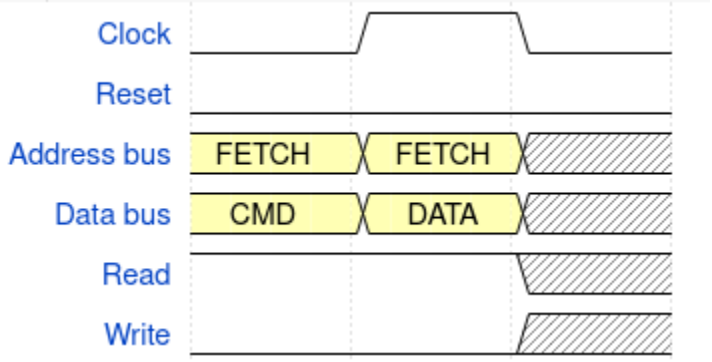
The interrupt is triggered on the next fetch cycle when: The interrupt disable bit is clear and the interrupt pin is pulled HIGH. When the interrupt is triggered, during the fetch cycle, the value 0x0000 (BRK) is force loaded into the instruction register, but the PC is not incremented. During an interrupt (or when executing a BRK instruction) an interrupt is not able to trigger again until the CPU leaves the interrupt state with a return from interrupt (RTI) instruction.

Timings

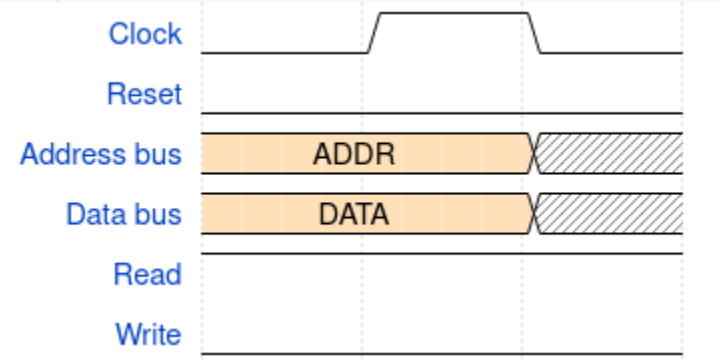
Reset cycle:



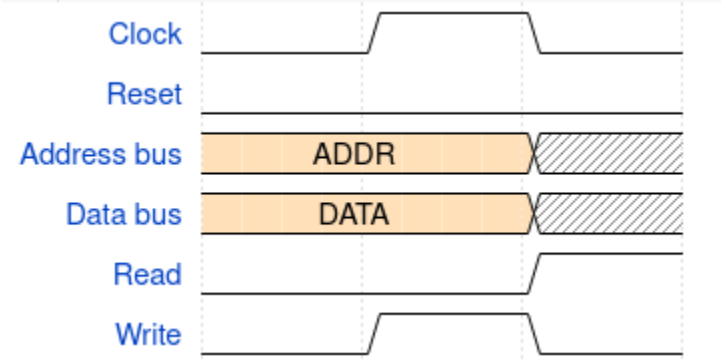
Fetch cycle:



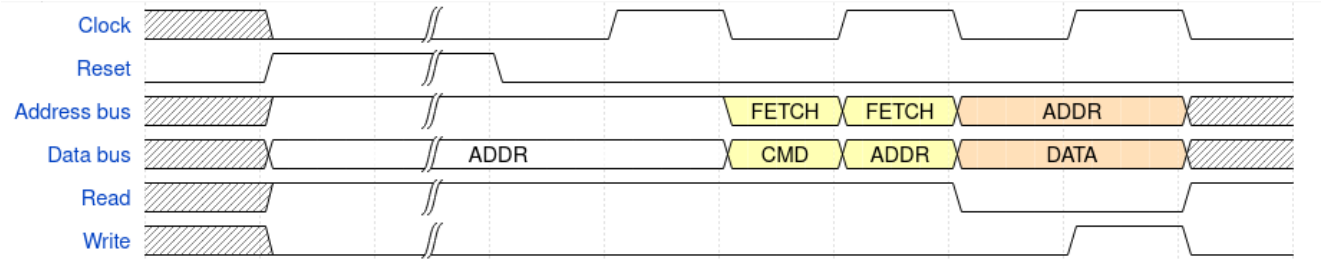
Execute cycle (Instruction that reads from memory):



Execute cycle (Instruction that writes to memory):



Full reset fetch and execute example (Instruction that writes to mem.):



PROGRAMMING

Instruction Set Architecture

Registers:

The LWC31 has 16 internal accessible registers. There are 13 general purpose registers and 3 special purpose registers.

Registers 1 to 13 are labeled r0-r12, r12 is also the result/remainder register (labeled rr) for DIV and MUL instructions, register 14 is a stack pointer (sp), register 15 is a status register (st), and register 0 is a special immediate data, read-only register. Register 0 is latched with immediate data on the second phase of the fetch cycle, this register is used like any other register to provide a flexible, but simple instruction set.

Instruction encoding:

Instruction are encoded in 16 bit opcodes, the opcode contains the instruction, operand 1, operand 2, and operand 3.

IIIIXXXXYYYYZZZZ

Least significant bit is on the right.

I (4 bits): Instruction
X (4 bits): Operand 1 register selector
Y (4 bits): Operand 2 register selector
Z (4 bits): General instruction options / register selector

Each instruction may use one, two or all three operands as inputs. Operand 1 and 2 are selectors to select which registers to use as inputs to the instruction, instructions that generates a result will be stored back to the register selected with operand 1, except the memory write instructions where operand 1 is the register used to write to memory. Operand 3 is used differently depending on the instruction. Operand 3 may be used as the index register for store and load instructions, ALU function selector, conditional

jump selector, or the operation selector for the interrupt option (INT) instruction.

Stack pointer:

The stack pointer register is a read and writable register that is automatically incremented/decremented when using the stack instructions (push, pop, jump to subroutine, return from subroutine, and return from interrupt). The SP register is a pre-decrement / post-increment model, meaning that when pushing to the stack, the stack pointer is decremented before writing to memory, and reads from memory before incrementing.

The stack pointer register is divided into 2 sections: latched, and working bits. The latch bits do not change when using stack operations, the latch is most significant 6 bits. The working bits are the least significant 10 bits, these bits change during stack operations. When writing to the stack register, all bits are written to (Latch and working bits), this gives a possibility of 64 sections of memory which can be allocated to a stack, and the stack may not move out of this set boundary.

Status register:

The status is a read and write register containing the flags that hold the CPU's state, the flags are: Carry, Zero, Negative, Interrupt disable, they correspond to bit 0, 1, 2, and 3 respectively. A carry is generated when shifting a bit out with shift functions, or when a carry occurs from an arithmetic function. The zero flag is set when a value written to a register is NULL. The negative flag is set when a value written to a register has the most significant bit set. Writing to this register will not cause Z or N bit to change/written with unexpected values. Bitwise operations on the status will not yield unexpected results on the Z/N bits, allowing successful setting/clearing of flags.

Instructions

The LWC31 has only 16 instructions, but due to the instruction encoding, every one of these instructions are powerful.

0x0 – BRK

BReaK enters an interrupt routine. When this instruction is executed, the CPU pushes the current PC to the stack and sets the PC to the address set with the INT instruction.

No operands.

0x1 – JMP

JuMP sets the PC to the value in operand 1.

Operand 1: Address to jump to

0x2 – MOV

MOVe copies contents from one register to another.

Operand 1: Destination register

Operand 2: Source register

Modifies: ZN

0x3 – LOD

LoaD from memory at the specified address.

Operand 1: Destination register

Operand 2: Address

Operand 3: If not null, this will specify an index register. (Effective address = OP2+OP3).

Modifies: ZN

0x4 – STO

STOre to memory at the specified address.

Operand 1: Source register

Operand 2: Address

Operand 3: If not null, this will specify an index register. (Effective address = OP2+OP3).

0x5 – ALU

Perform an ALU operation on two registers and store the result back to the first operand. If the ALU function is bit shifting, operand 2 is ignored.

Operand 1: Data

Operand 2: Data

Operand 3: ALU function

Modifies: CZN (Does not modify the carry on bitwise operations and the negate operation)

ALU functions:

0x0 – ADD (Add)

0x1 – ADC (Add with carry)

0x2 – SUB (Subtraction)

0x3 – SBC (Subtract with carry)

0x4 – SHL (Shift left bits in operand 1 by one bit, MSB is shifted into the carry flag.)

0X5 – ROL (Same as SHL, except the carry flag is shifted in to the LSB)

0x6 – SHR (Shift right bits in operand 1 by one bit, LSB is shifted into the carry flag.)

0X7 – ROR (Same as SHR, except the carry flag is shifted in to the MSB)

0x8 – AND (Bitwise AND)

0x9 – OR (Bitwise OR)

0xA – XOR (Bitwise Exclusive OR)

0xB – MUL (Unsigned multiply)

Upper 16 bits of result stored in r12 (rr)

0xC – SMUL (Signed multiply)

Upper 16 bits of result stored in r12 (rr)

0xD – DIV (Unsigned divide)

Remainder stored in r12 (rr)

0xE – SDIV (Singed divide)

Remainder stored in r12 (rr)

0xF – NEG (Negate / Generate 2's compliment)

0x6 – CMP

CoMPare two values. Perform a non-storing subtraction of OP1 and OP2 and set the flags corresponding to the results.

Operand 1: Data

Operand 2: Data

Modifies: CZN

0x7 – JIF

Jump IF condition specified in operand 3 is true.

Operand 1: Address to jump to

Operand 3: Bitwise AND with the first three bits of the status register. If the result is non-zero, PC is set to operand 1. MSB of operand 3 will invert the condition.

0x8 – NOP

No Operation. Waste a clock cycle

0x9 – INT

INTerrupt options. This instruction can be used to set and disable the interrupt disable bit, or it can be used to set the interrupt vector.

Operand 1: Interrupt vector. This is used if operand 3 is NULL.

Operand 3:

0x0: Set interrupt vector

0x1: Set interrupt disable

0x2: Clear interrupt disable

0xA – PLP

PuLl Processor status. This instruction pops a value off the stack and writes it into the status register.

0xB – PUSH

PUSH data onto the stack.

Operand 1: Data

0xC – POP

POP data off the stack and write it to a register.

Operand 1: Destination register

Modifies: ZN

0xD – JSR

Jump to SubRoutine. Push the current PC to the stack and set the PC to operand 1.

Operand 1: Address

0xE – RTS

ReTurn from Subroutine. Pop the return address from the stack and write it to the PC.

0xF – RTI

ReTurn from Interrupt. Pop the return address from the stack and write it to the PC, and exit from the interrupt state.