QUANTUMPULSE™ CONTROL ASSEMBLAGE 77B

QPCA-77B REFERENCE MANUAL

SYSTEM OVERVIEW

The QuantumPulse™ Control Assemblage QPCA-77B is an exciting new state-of-the-art computing platform! The system comprises of three slots that can be filled with AstroWave™ computing nodes of various sizes. Each node operates independently but perfectly synchronized due to our patented QuantumPulse technology. Though each node is independent, we all know the real work gets done by collaboration! Note that for quantum observability reasons, nodes cannot make decisions based on their own state but can only respond to the state broadcast by other nodes.

1. NODE OPTIONS

AstroWave™ computing nodes all share a number of common properties, and only vary in how many of each feature they have. All nodes have:

- 1 **accumulator** (***ACC***) register that can be used for temporary storage as well as incremented and decremented
- 4 or more **lines of code** each of which can contain a label, an operation, and a comment
- 1 or more **radios** which are automatically enabled when you send data to a specific channel

Your **QPCA-77B** starter kit comes equipped with a supply of AstroWave™ nodes with the following capacities:

- **AW0401**: 4 **lines of code** and 1 **radio**
- **AW0903**: 9 **lines of code** and 3 **radios**
- **AW1605**: 16 **lines of code** and 5 **radios**

2. COMMUNICATION CHANNELS

Nodes can communicate with each other by using one of their radios to broadcast on one of limitless communications channels [1]. Any time a node writes a value to a channel, a radio is allocated and the node continues to broadcast the value until the node writes a different value, or a 0 to disable broadcasting.

Notes:

- If a node is broadcasting to specific channel, it cannot read from that channel in any way until after it has stopped broadcasting (by writing a 0 to that channel)
- If two or more nodes are broadcasting on the same channel, any receiver will read their combined (summed) value
- Though channels are theoretically limitless, the QuantumPulse™ Visualization Interface will only show you the values being broadcast on channels CH1 through CH14
- AstroWave node execution is synchronized at *faster than light speed* [1]! If one node reads from a channel on the same cycle that another node starts broadcasting or changes it's value, the reading node will receive the value that was being broadcast on the previous cycle. Note: Our *Quantum Innovations* department assures us of the inaccuracy of our competitors' reductive claims that "reading from a radio channel obtains the value being broadcast on that cycle, and writing to a channel simply causes the written value to be transmitted starting on the following cycle."

[1] This statement is classified by the *Truth in Advertising Act* of 1972 as *advertising-truth*.

(this page unintentionally left blank)

3. NODE EXECUTION

Nodes execute their operations in sequential order, and execution automatically returns to the beginning after executing the final instruction. All operations take exactly one cycle.

3.A. ANATOMY OF A LINE OF CODE

Example: `foo: MOV ACC INPUT ; bar!`

- `foo:` is a label, used as a reference by JUMP instructions
- `MOV` is the instruction (see below)
- `ACC INPUT` are the operands to the instruction (specifying registers or values, varies by instruction)
- `; bar!` is a comment (ignored by the processor)

3.B. REGISTERS

3.B.1 ACC

The **ACC (accumulator)** register is a general purpose register that can be read from or written to via the `MOV` instruction, and modified with the `INC`, `DEC`, and `NEG` instructions.

3.B.2 INPUT

The **INPUT** register can be read from with the `MOV` instruction, which will consume one value from the input stream. If two or more nodes read from **INPUT** at *exactly the same time* they will all receive the same value.

3.B.3 OUTPUT

The **OUTPUT** register can be written to with the `MOV` instruction. Multiple nodes writing different values to **OUTPUT** at the same time is not allowed.

3.B.4 RADIO CHANNELS CH1 ... CHX

Channels can be written to or read from with the `MOV` instruction, as well as used as a source of comparison for the **JUMP** family of instructions

3.B.5 NIL

Writing to the **NIL** register does nothing. Reading from the **NIL** register is the same as using the literal number 0.

3.B.6 ACCEPTED VALUES

All values in registers will be clamped between negative and positive Biblical Infinity (seventy times seven).

3.C. INSTRUCTIONS

3.C.1 MOV DESTINATION SOURCE

Reads from **source** and writes the read value to **destination**. The source can be any readable register, or a literal number. The destination must be a writable register. Examples:

- `MOV ACC INPUT`
- `MOV OUTPUT ACC`
- `MOV ch7 99`

3.C.2 DEC AND INC

These decrement or increment the value stored in the **accumulator (ACC)** register.

3.C.3 NEG

This negates the value stored in the **accumulator (ACC)** register (multiplies it by negative 1).

3.C.4 NOP

This is shorthand for `MOV ACC ACC` which does nothing except take 1 cycle of time.

3.C.5 JMP LABEL

Moves execution to the first instruction after the specified label instead of continuing to the next instruction.

3.C.6 JUMP FAMILY - JLZ / JGZ / JEZ / JNZ - J*Z CH# LABEL

Compares the value read from the specified channel against 0 and jumps to the specified label if the comparison succeeds.

- JLZ - Jump if less than 0
- JGZ - Jump if greater than 0
- JEZ - Jump if equal to 0
- JNZ - Jump if not equal to 0

For convenience, a number, or readable register, can be used instead of a label, and the jump will be relative. For example JMP -1 will jump to the preceding line.

Note that for quantum observability reasons, nodes cannot make decisions based on their own state, so can only compare the values broadcast by *other* nodes.

4. TRAINING EXERCISES

The QuantumPulse™ Visualization Interface contains a series of training exercises that will help you master your craft! Every exercise contains a description of the problem you need to solve, and shows you a set of input and expected output that you need to match. To pass an exercise, your solution needs to solve the provided input/output as well as 2 more input/output sets which will only appear after you've solved the first ones. It does not matter what your solution does after writing the final correct output value.