**Compiler**

* Syntax similar to AMORAL,but entirely interpreted.
* Split up into procedures w/parameters.
* Replace all strings with constants holding addresses.
* Replace all constants (various formats) with addresses.
* Replace all variables with addresses. $ is used to prefix locals.
* Centralised storaged for constants & strings to minimise duplication.
* Compile code for lall procedures.
* Align to byte page.

**Memory Usage**

|  |  |
| --- | --- |
| **Runtime**  **binary** | Header (64 bytes):   * Jump to initial code (+0) * Jump to opcode executor (+4) * Address of first procedure (+8) * Load address (+10) * Start of Variable space to erase (+12) * End of Variable space to erase (+14) * Address of Register (+16) * Next Free Zero Page Address (+18) * Linked list of known routines (+20) |
| Runtime code interpreter, standard routines. |
| **Libraries** | Library code written in 6502 assembler. Uses a FORTH type link but offset forward so can be simply battened together which has name, param count, address of 1st parameter (if > 1), as usual last parameter is passed in R. |
| **Compiler**  **Generated**  **Binary** | Generated code from compiling MTL |
| ASCIIZ Strings. |
| System variables (e.g. free.memory) *starts on page boundary*. |
| Variables containing constant values. |
| **Free**  **Memory** | Variable space (erased to zero on start up) |
| Free memory space. |

**Runtime**

|  |  |
| --- | --- |
| **Opcodes** | **Instructions** |
| 0000-8FFF | LDR STR ADD SUB MUL DIV AND ORR XOR, the operand is a variable index into the variable space (e.g. doubled and added to the variable base address which is on a page boundary). Opcode is 12 bit unsigned. |
| 9000-BFFF | BRA BEQ BPL conditional branch which conditional on the current value. Opcode is 12 bit signed. |
| C000-FFFF | CALL a 6502 routine at (lower 12 bits x 4). Routines executing runtime will start with a call to the runtime executor (stored in vectors) |