**MAHCPU Sprache - Dokumentation**

|  |
| --- |
| **Infos:** |
| Jeder Wert kann auch der Wert einer Variable sein, wenn $ mit dem Variablennamen steht.  Benenne bitte nichts „SYS\_...“ | |

**1) Variablen:**

**1) Erstellung und Setzung:**

|  |
| --- |
| **new\_var** |
| **new\_var (Variablenname)** | |
| Der Befehl new\_var erstellt eine neue Variable im Speicher und setzt diese auf 0. | |

|  |
| --- |
| **set\_var** |
| **set\_var (Variablenname) (Wert)** | |
| Der Befehl set\_var setzt eine Variable auf einen Wert. | |

**2) Mathematik:**

|  |
| --- |
| **m\_add\_var** |
| **m\_add\_var (Variablenname) (Wert)** | |
| Der Befehl m\_add\_var setzt eine Variable auf das Ergebnis vom Wert der Variable + einem Wert. | |

|  |
| --- |
| **m\_rem\_var** |
| **m\_rem\_var (Variablenname) (Wert)** | |
| Der Befehl m\_rem\_var setzt eine Variable auf das Ergebnis vom Wert der Variable - einem Wert. | |

**3) Logik:**

|  |
| --- |
| **m\_and\_var** |
| **m\_and\_var (Variablenname) (Wert)** | |
| Der Befehl m\_and\_var setzt eine Variable auf das Ergebnis von AND vom Wert der Variable und einem Wert. | |

|  |
| --- |
| **m\_or\_var** |
| **m\_or\_var (Variablenname) (Wert)** | |
| Der Befehl m\_or\_var setzt eine Variable auf das Ergebnis von OR vom Wert der Variable und einem Wert. | |

|  |
| --- |
| **m\_not\_var** |
| **m\_not\_var (Variablenname)** | |
| Der Befehl m\_not\_var setzt eine Variable auf das Ergebnis von NOT vom Wert der Variable | |

**2) I/O:**

**1) VRAM:**

|  |
| --- |
| **set\_vram** |
| **set\_vram (Wert) (Wert)** | |
| Der Befehl set\_vram setzt ein Element des VRAMs auf einen Wert. Der VRAM besteht aus 3 Elementen pro Pixel (RGB). (Es beginnt bei 0)  Der Bildschirm besteht aus 100x100 Pixeln | |

|  |
| --- |
| **set\_pixel** |
| **set\_pixel (Wert) (Wert) (Wert) (Wert)** | |
| Der Befehl setzt einen Pixel auf dem VRAM auf die drei Werte (im RGB Format).  (Der erste Wert, ist der Index. Der Index wird sollte beim R Index des Pixels sein => 2. Pixel => Index = 3(R Wert des Pixels)) | |

**2) Bildschirm:**

|  |
| --- |
| **print\_scr** |
| **print\_scr** | |
| Der Befehl print\_scr kopiert alle Daten im VRAM auf den eigentlichen Bildschirm und zeigt somit das Bild an. | |

|  |
| --- |
| **draw\_img** |
| **draw\_img (Dateipfad/Name)** | |
| Der Befehl draw\_img kopiert ein 100x100 Bitmap-Bild in den VRAM-Speicher und zeichnet es anschließend. | |

**3) Eingabe:**

|  |
| --- |
| **inp** |
| **inp (Variablenname)** | |
| Der Befehl inp setzt eine Variable auf den Unicode Wert des eingegebenen Zeichens. Nach der Eingabe des Zeichens wird aber nicht gewartet und der nächste Befehl wird direkt ausgeführt. | |

|  |
| --- |
| **input** |
| **input (Variablenname)** | |
| Der Befehl input setzt eine Variable auf den Unicode Wert des eingegebenen Zeichens. Es wird gewartet, bis ein anderes Zeichen gedrückt wird. | |

**3) Jumps und If:**

**1) Jumps:**

|  |
| --- |
| **:** |
| **: (Name)** | |
| Definiert einen Jump-Punkt.  Man kann mit jump zu diesem Punkt springen | |

|  |
| --- |
| **jump** |
| **jump (Jumpname)** | |
| Der Befehl jump springt zu einem Jump-Punkt. | |

**1) Ifs:**

|  |
| --- |
| **if\_jump** |
| **if\_jump (Jumpname) (Variablenname)** | |
| Der Befehl if\_jump springt zu einem Jump-Punkt, wenn der Wert der Variable NICHT 0 ist. | |

**4) Debug:**

|  |
| --- |
| **debug\_prnt** |
| **debug\_prnt (Name)** | |
| Der Befehl debug\_prnt gibt den Wert einer Variable in der Konsole aus. (Da der CPU eigentlich keine Konsole hat, ist der Befehl nur für Emulatoren zum Testen geeignet!) | |