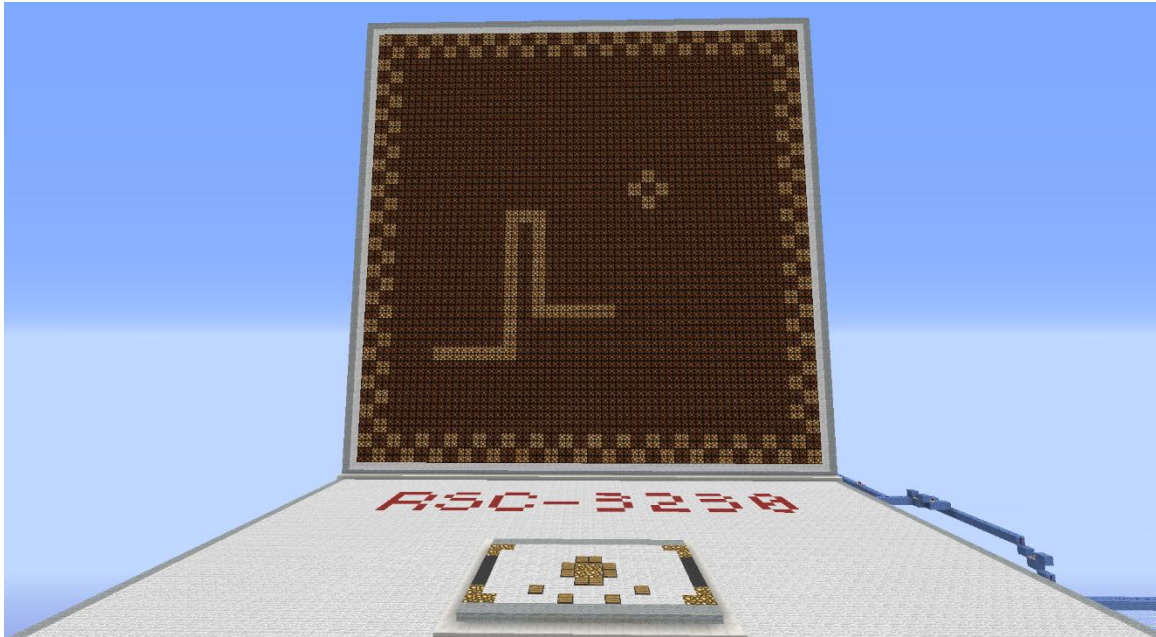


RSC-3230 使用手册

V1.1



本作品是一台 32 位红石电脑，CPU 部分是一个 32 位通用处理器，配有 512Byte RAM（其中 128Byte 为显示缓冲区），8 键输入（其中四个方向键，四个功能键），32x32 像素屏幕输出。

作者：HappyWater@TRP

2016 年 4 月

一、作品总览

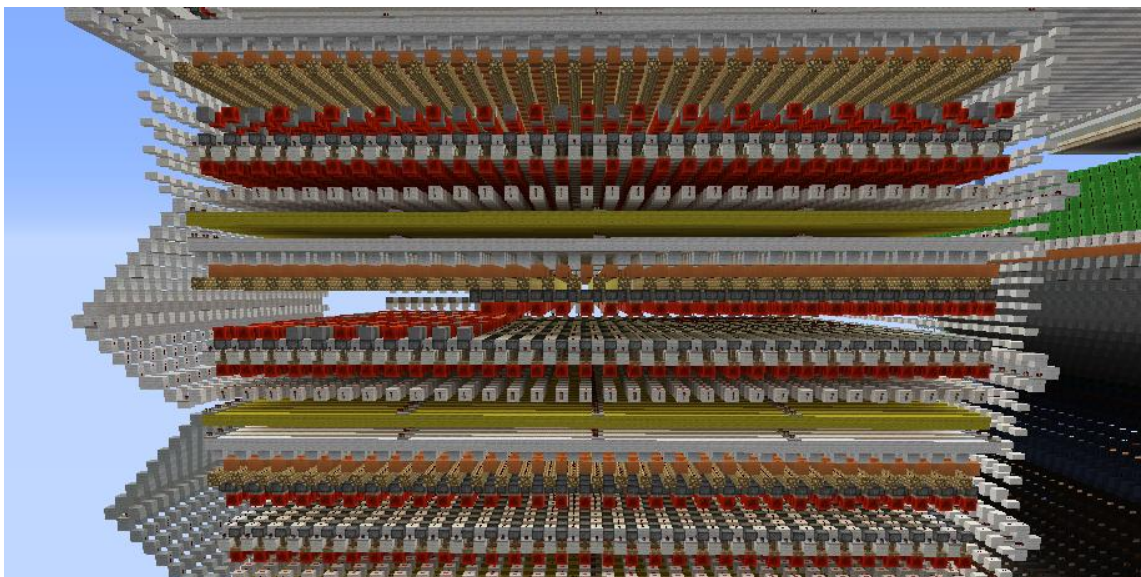
本作品总体由运算器模块，存储器模块，控制器模块，显示屏模块，以及输入按钮等部分组成。下面简要介绍各个模块。

1、运算器



运算器包含一个加法器，两个单向串行移位器（分别对应左移和右移），一个串行乘法器和一个串行除法器。运算器组可以完成加减乘除四则运算，与或非逻辑，移位运算等运算。

2、存储器



存储器总共 128 个字，每个字长 32 位，不允许非对齐访问。建筑结构为多层结构，总共分四层，每层容量 128Byte，其中最上面一层被连接在显示屏上，使用屏幕刷新指令即可将这部分存储器中的数据 displays 在屏幕上。

存储器中，靠下的层为地址低位，靠上的为高位。

3、控制器



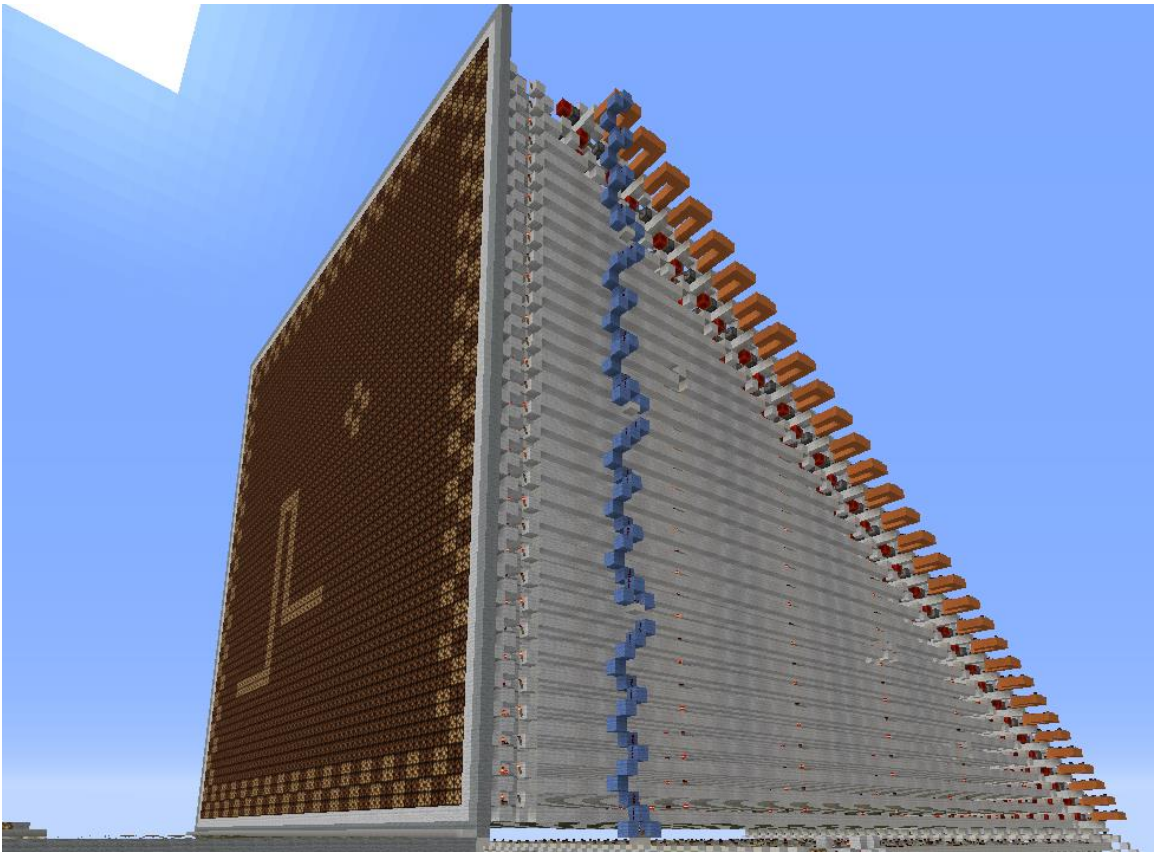
控制器部分负责产生控制时序，指挥整机的运转。控制器也是分层结构的，面板上每一行对应内部的一层。每一层负责一种基本动作，按下每一层右边的按钮，即可执行这一层的动作。

在正常自动运行时，控制器会依照指令，轮流执行相应的动作，完成程序预期的操作和运算。在单步运行时，玩家可以手动执行每一个动作，观察程序运行状态，以便调试程序。

下数第四层，也就是橙色的一层为取指动作层。通常情况下，按下这一层的启动按钮即是启动整台机器。

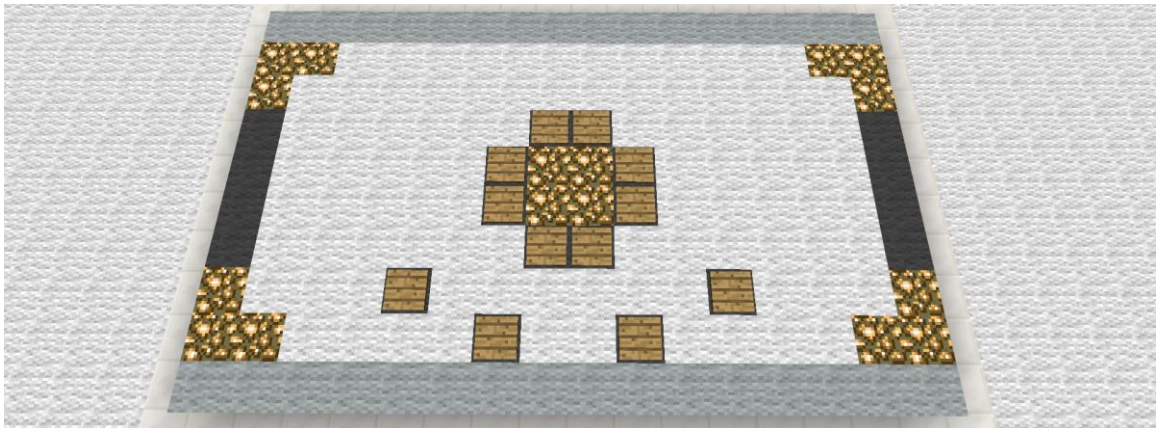
在面板右侧有一个拉杆，是整机的停机拉杆。当退出游戏时，应先停止机器的运行。**MC**在保存地图时，对各个部分的保存时间并不同步，若此时机器处于运行状态，则可能导致红石电脑出错，甚至处于失控状态。

5、显示屏



显示屏为 32x32 像素红灯屏幕。利用活塞 **BUD** 暂存屏幕画面，当程序正在绘制显示缓冲区中的画面时，显示屏上的画面不受影响，当绘制完成后，可使用刷新屏幕指令将画面刷新到屏幕。

6、输入按键



本作品提供了 8 个输入按键，如图中所示，中间是四个方向键，下面四个是功能键。

8 个按键直接连接在寄存器 R30 上，软件访问寄存器 R30 即可获取当前按键状态，对 R30 写入则无意义。R30 最高 8bit 中，每一个 bit 对应一个按键，从最高位（31 位）向低位依次为“上”“右”“左”“下”四个按键，紧接着下面从左向右依次四个功能按键。

当按键没有按下时，对应 bit 位为 0，按下时对应比特位为 1。R30 中其余 bit 位没有连接外部设备，可留给玩家自己 MOD 添加。按键按下后并不会对系统产生中断，按键信息的输入只能通过软件轮询实现。

二、指令表

本 CPU 的指令为 32 位定长，在内存中对齐存放。内存访问时忽略地址的最低 2bit。

CPU 中包含 32 个 32 位通用寄存器，其中 R31 为指令计数器，R30 为键盘状态寄存器。当指令运算中引用 R31 时，R31 永远指向当前指令的下一条指令地址。当指令执行结果修改 R31 时，程序流程即跳转。

1、数据传送指令

单寄存器指针内存读取指令：LDRs R1,R2

即以寄存器 R1 中的数值为指针访问内存，将内存中的数据传送寄存器 R2 中。两个寄存器编号可以相同。

双寄存器指针内存读取指令：LDRd R1,R2,R3

即以寄存器 R1、R2 之和为指针访问内存，将内存中的数据传送到寄存器 R3 中。三个寄存器编号可以交叉。

单寄存器内存写入指令：STRs R1,R2

道理同上，以 R1 为指针，将 R2 写入内存。

双寄存器指针内存写入指令：STRd R1,R2,R3

道理同上，以 R1、R2 之和为指针访问内存，将 R3 写入内存。

立即数输入指令：LDRi R1, \$xx

即向寄存器 R1 中输入一个有符号整数立即数。该立即数长度 19 位，符号扩展至 32 位存入 R1。

上述指令编码如下：

LDRs STRs LDRd STRd

0000 0100 1000 1100 => AAAA

[11][AAAA][00][R1][R2][R3][000000000] 左边 msb , 右边 lsb。

LDRi R1, \$int

[11][0010][00][R1][int]

例：

LDRs R1, R2 11 0000 00 00001 00000 00010 000000000

STRd R4, R5, R6 11 1100 00 00100 00101 00110 000000000

LDRi R16 , \$12345 11 0010 00 10000 0000011000000111001

LDRi R7 , \$ -19 11 0010 00 00111 1111111111111101101

2、算术运算及逻辑运算指令

本机器中 ALU 支持 ADD SUB OR NOR AND NAND XOR NXOR 等指令，指令结构相似。

ADD R1, R2, R3 即将 R1, R2 中的数值相加，结果存入 R3 中。三个寄存器的编号可以交叉。

SUB R1, R2, R3 即将 R2-R1 结果存入 R3。三个寄存器编号可以交叉。

OR、NOR、AND……等其余指令类似不再赘述。

本机中移位器支持逻辑左移（BSL）、循环左移（BCL）、逻辑右移（BSR）、循环右移（BCR）以及算术右移（BAR）等指令。

BSL R1, R2, R3 即将 R1 中的数值逻辑左移 R2 位，将结果存入 R3。三个寄存器编号可以交叉。R2 中的数值忽略高位，仅关注最低 5bit。

BCL、BSR、BCR、BAR 等指令同理，不再赘述。

本机中乘法器和除法器分别完成乘法和除法指令。需注意的是，乘法器与除法器指令涉及 4 个寄存器，两个源操作数，两个目的操作数。

MUL R1, R2, R3, R4 即将 R1 R2 中的数值相乘，结果存入 R3 R4 中。其中 R3 存放结果低位，R4 存放结果高位。四个寄存器的编号可以交叉，若 R3 R4 编号相同，则最终寄存器中只存放结果低位。

上述指令编码如下：

[00][AAAA][F][C][R1][R2][R3][R4][A][000] （注意：A 字段有前后两组，共 5bit）

ADD SUB OR NOR AND NAND XOR NXOR

00000 01000 00001 00101 11101 11001 00110 00010 => AAAAA

（本组指令可影响标志位）

BSL BSR BCL BCR BAR MUL DIV

00011 10011 01011 11011 10111 10000 10100 => AAAAA

（对本组指令使用标志影响位结果将不可预知）

指令中，F 和 C 各占 1 个 bit。利用这两个指示位可实现 if goto 指令。

F 为标志影响位，若为 1，则本指令执行结果将影响标志位，若运算结果为 0，则标志位为 1，反之标志位为 0。

C 为条件执行位，若本为 1，则本指令在标志位为 1 时才执行，否则将跳过不执行。若本位为 0，则本条指令无条件执行。

3、屏幕控制指令

清白屏：WS 011000[其余填零]

清黑屏：BS 010100[其余填零]

刷新屏：RS 010010[其余填零]

WS 和 BS 用于清屏，区别是 WS 让全屏变亮，BS 让全屏变暗。由于显示屏利用活塞 BUD 暂存画面，有时活塞可能“失去粘性”，此时应使用清白屏指令使其恢复。

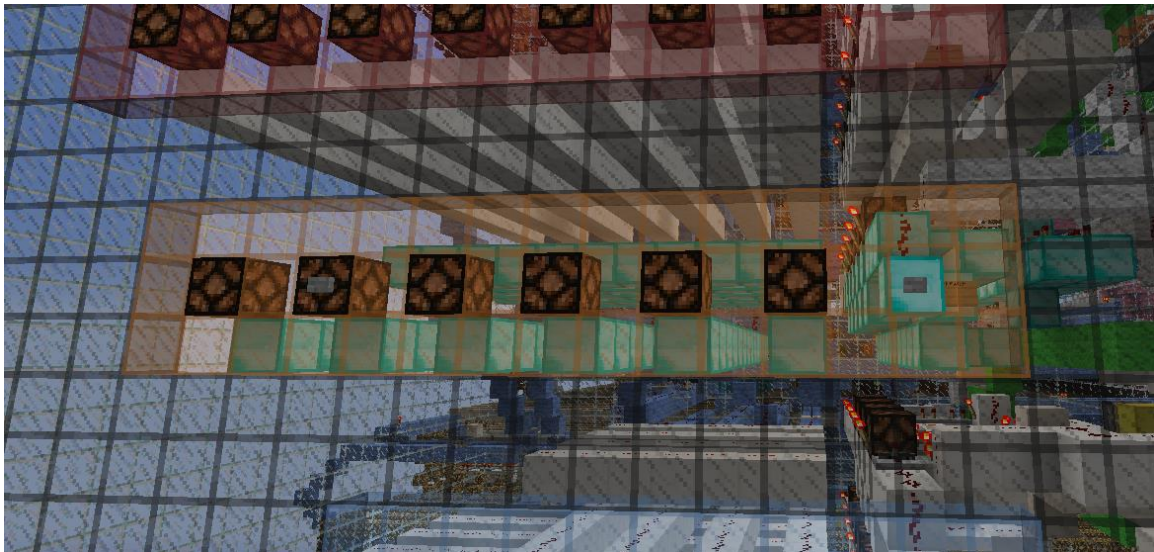
刷新屏指令将 RAM 中显示缓冲区里的数据显示在屏幕上。

显示缓冲区为 128 字节，每 4 字节对应屏幕上一行，高位对应靠左的像素，低位对应靠右的像素。靠高地址端的对应屏幕底下的像素，靠低地址端的对应屏幕上方的像素。

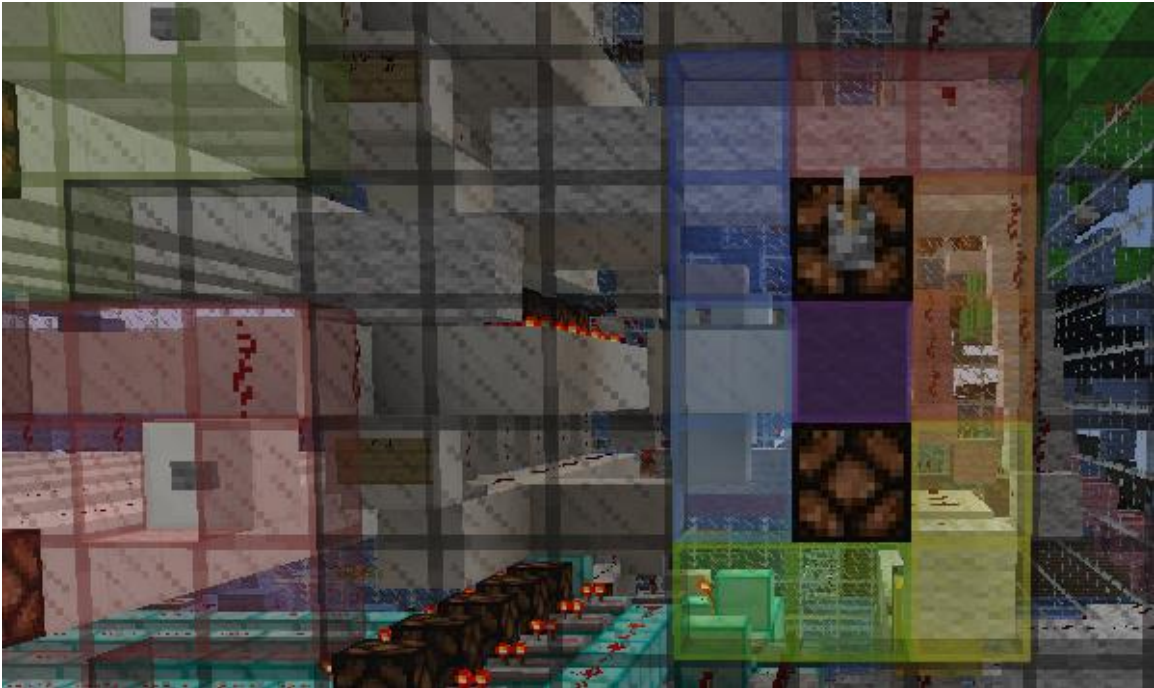
三、使用方法简述

本机未设置 ROM，整个存储器部分都为 RAM，既可存放指令，亦可存放数据。可手动修改每一个存储位状态以完成编程。若不希望指令被意外修改，可将存储区相应的存储位上的活塞去除，以固定存储位状态。

在机器处于停止状态时，按下下图中左数第二个灯上的按钮即可将 R31 清零，以完成程序复位。软件的第一条指令应按放在内存中的第一个单元内。可以在第一个内存单元处安置一条跳转指令，即可跳跃至任意程序入口。软件亦可在各个寄存器中存放初始化数据，而不必各个都从 RAM 中加载，以节省 RAM 空间和运行加载时间。



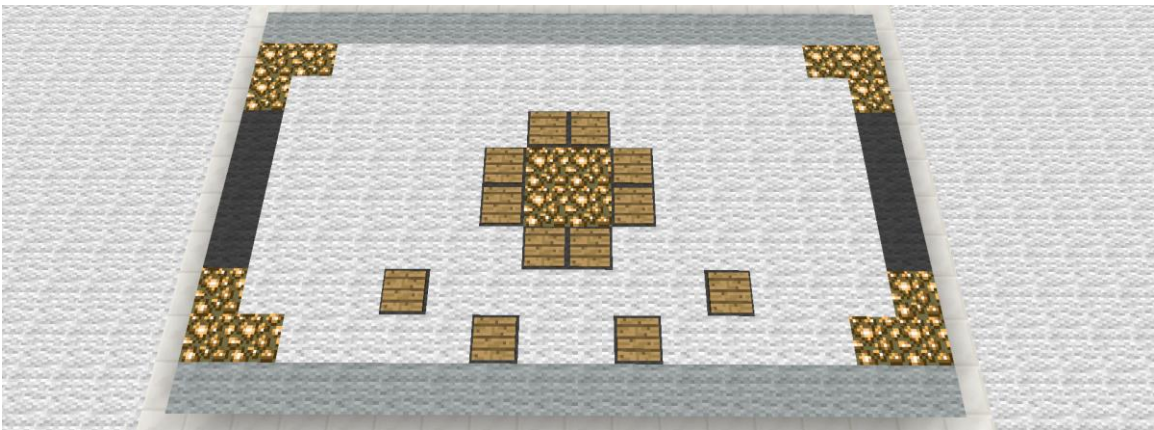
如上图所示，控制器面板中橙色的一层为指令读取动作层。按下右边的按钮，机器将从 R31 所指向内存单元读入指令并存入指令暂存器中，然后立即跳跃到对应的动作层去执行该指令。



如图所示，图中的拉杆为停机拉杆。任意时刻，若拉杆充能，则当前指令执行完毕后，整机便停止运行，机器停止时，左边的指示灯会闪烁一次。

在拉杆充能的情况下，按下取指令动作的按钮，则机器将完成取指和执行两个动作后停止运行。此方法可适用于程序单步执行。

若按下其它各层动作指令按钮，则将按各层方式解析当前指令并执行。每条指令只可以对应一个动作层，若启动的动作层与当前指令不匹配，则结果将不可预知。



屏幕面前地板上的按键连接至寄存器 R30 上，软件需轮询 R30 以完成按键输入，无按键中断功能。