perations			128	64	32	16	8	4	2	1
nmediate	imm		0	0						
alculate	calc		0	1						
ору	сру		1	0						
Condition	cond		1	1						
egister 0 in	r0i				0	0	0			
egister 1 in	r1i				0	0	1			
egister 2 in	r2i				0	1	0			
egister 3 in	r3i				0	1	1			
egister 4 in	r4i				1	0	0			
egister 5 in	r5i				1	0	1			
put	in				1	1	0			
nused					1	1	1			
egister 0 out	r0o							0	0	0
egister 1 out	r1o							0	0	1
egister 2 out	r2o							0	1	0
egister 3 out	r3o							0	1	1
egister 4 out	r4o							1	0	0
egister 5 out	r5o							1	0	1
utput	out							1	1	0
nused								1	1	1
r	or							0	0	0
and	nand							0	0	1
or	nor							0	1	0
nd	and							0	1	1
dd	add							1	0	0
ub	sub							1	0	1
nused								1	1	0
nused								1	1	1
ever	never							0	0	0
qual to 0	jeq							0	0	1
ess than 0	jlt							0	1	0
	jle							0	1	1
ways	jmp							1	0	0
ot Equal to 0	jne							1	0	1
reater than or Equal to 0								1	1	0
	jgt							1	1	1
struction Layouts										

_											
Сору	128	64		16			4	2	1		
	Operation		Copy from			Copy to					
					_			_			
Calculation	128	64		16	8		4	2	1		
	Operation		Unused			Condition					
Condition	128	64		16			4	2	1		
	Operation		Unused			Function					
Immediate	128	64		16	8		4	2	1		
	Operation		Value								
Info t Oak											
Info + Setup											
	"Immediate" is the term for as	nding a value dire	othy to road. The in	nmadiata valua ia	marked in the hite	1 6 of the inc	ruotion bu				
	"Immediate" is the term for se		city to <u>regu</u> . The ir	nmediate value is	marked in the bits	i i-o oi the ins	ruction by	e.			
	An immediate value can rang	9 110111 0 - 63									
	Calculations always take the	values in real and	roa? and do the	function on thom	Pog1 is on the left	of the calcula	tion				
	Example: Addition = reg1 + re		-	direction on them.	rteg i is on the len	or trie calcula	lion				
	Example: Addition = reg1 1 re	gz, Subtraction – i	egi - leg z								
	Conditions always take the va	lue in red3 and co	mnare it against tl	ne aiven condition							
	Conditions always take the value in <u>reg3</u> and compare it against the given condition. If the condition is evaluated as true, the program counter will jump to the line of code at the value stored in <u>reg0</u> .										
	27.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.										
	When programming, the furthest left bit is the 128th bit. To program a component, simply replace low constants with high constants and rewire.										
	When finished, package the v		· -				J				
	When you have added it into the file, rewire it in place of the placeholder program component.										
	,			, ,							
Setup											
	Toggle the PRE and CLR switches by the program component and replace the program component with your own program.										
	The processor must be hand-pulsed, and pulses will be controlled by the switch labelled "".										
	When you tick the processor, always wait for a small amount of time before tocking it, so as to avoid any unwanted errors.										
	The "Input block" is where you	set the input valu	e. It shows both the	ne binary, and hex	representation of	the number.					
	The "Output block" is where the	ne final result will b	e sent, that is if yo	ou actually prograi	m it to do so.						
	Enjoy!										
Assembly Code Syntax											
Using the Replit file	Сору:	cpy (from adr) (to	o adr)								
	Calculate / Condition:	calc / cond (oper	ration)								
	Immediate:	imm (value)									

Conditions are slightly tricky to implement. Because the jump address is always stored in r0, you must send an immediate value to r0 before the jump.									
The immediate value should be the byte of code that you want to jump to, e.g. byte 2, etc.									
Your lines of code begin at byte 0, and increment by 1 for each instruction.									
After the immediate instruction, you then add your jump condition, e.g. cond jne.									
In the Replit program, there is a multiline input for a code block. When you have finished your code block, simply type 'end' to convert it.									