

# Real Time Logic Simulator

Jari Komppa  
March 27, 2008

# Outline

- Development of the idea
- Development process
- The result

- Original idea: graphical UI to my 8051 simulator (which is a console application)

```
C:\vcproj\emu8051\core.exe
m>Low
0000 00 00 00 00 00 00 00 00
0008 00 00 00 00 00 00 00 00
0010 00 00 00 00 00 00 00 00
0018 00 00 00 00 00 00 00 00
0020 00 00 00 00 00 00 00 00
0028 00 00 00 00 00 00 00 00
0030 00 00 00 00 00 00 00 00
0038 00 00 00 00 00 00 00 00

Low0000: 0 0 0 0 0 0 0 0
Cycles :      72
Time   :      0.006ms
HW     : Super8051 @12.0MHz

Stck SP-P0-P1-P2-P3-IP-IE C-ACF0R1R00v--P
00 07 FF FF FF FF 00 82 0 0 0 0 0 0 0 0
00 07 FF FF FF FF 00 82 0 0 0 0 0 0 0 0
00 07 FF FF FF FF 00 82 0 0 0 0 0 0 0 0
00 07 FF FF FF FF 00 82 0 0 0 0 0 0 0 0
00 07 FF FF FF FF 00 82 0 0 0 0 0 0 0 0
00 07 FF FF FF FF 00 82 0 0 0 0 0 0 0 0
00 07 FF FF FF FF 00 82 0 0 0 0 0 0 0 0
00 07 FF FF FF FF 00 82 0 0 0 0 0 0 0 0

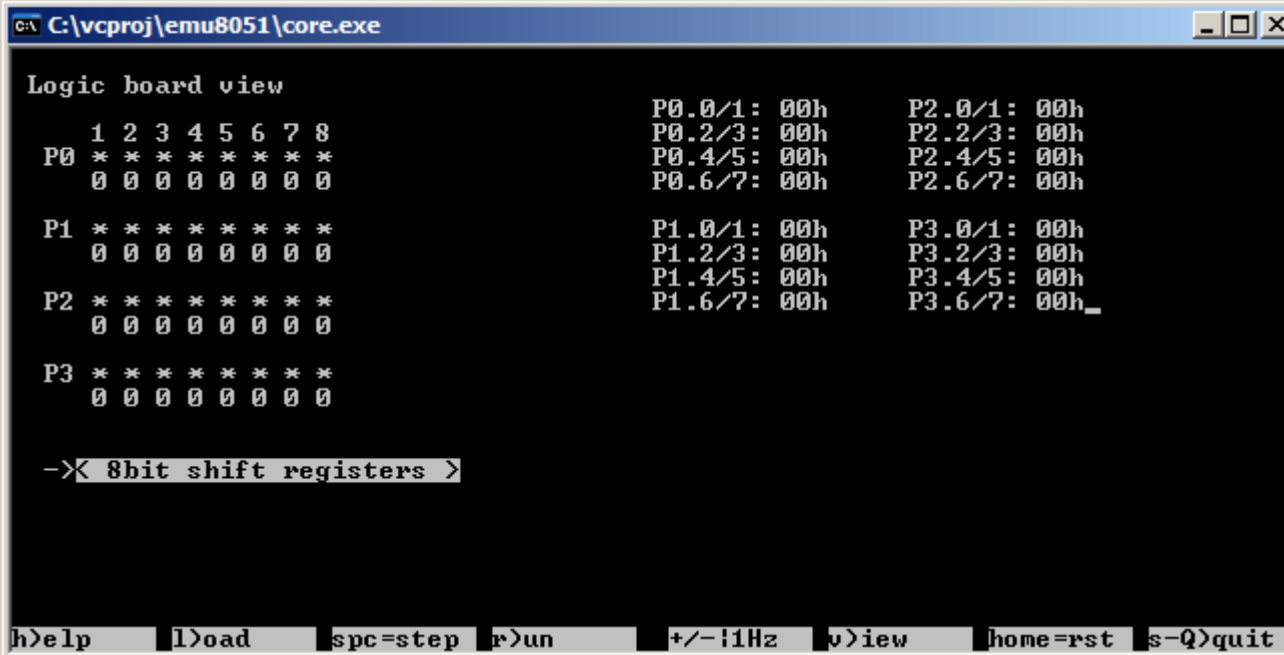
TMOD-TCON--TH0-TL0--TH1-TL1--SCON-PCON
0A 10 EB ED 00 00 00 00
0A 10 EB ED 00 00 00 00
0A 10 EB ED 00 00 00 00
0A 10 EB ED 00 00 00 00
0A 10 EB ED 00 00 00 00

PC Opcodes Assembly A -R0-R1-R2-R3-R4-R5-R6-R7-B -DPTR
0000 02 01 00 LJMP #0100h 00 00 00 00 00 00 00 00 00 00 0000
0100 D2 A9 SETB IE.1 00 00 00 00 00 00 00 00 00 00 0000
0102 D2 8C SETB TCON.4 00 00 00 00 00 00 00 00 00 00 0000
0104 75 8A EB MOV TL0, #EBh 00 00 00 00 00 00 00 00 00 00 0000
0107 75 8C EB MOV TH0, #EBh 00 00 00 00 00 00 00 00 00 00 0000
> 010A 75 89 0A MOV TMOD, #0Ah 00 00 00 00 00 00 00 00 00 00 0000

h>elp |l>oad |spc=step |r>un |+/-|Hz |v>iew |home=rst |s-Q>quit
```

- <http://iki.fi/sol/8051.html>

- The original simulator had a 'logic board' view – LEDs, buttons, other peripherals..



```
C:\vcproj\emu8051\core.exe

Logic board view

   1 2 3 4 5 6 7 8
P0 * * * * * * * *
   0 0 0 0 0 0 0 0

P1 * * * * * * * *
   0 0 0 0 0 0 0 0

P2 * * * * * * * *
   0 0 0 0 0 0 0 0

P3 * * * * * * * *
   0 0 0 0 0 0 0 0

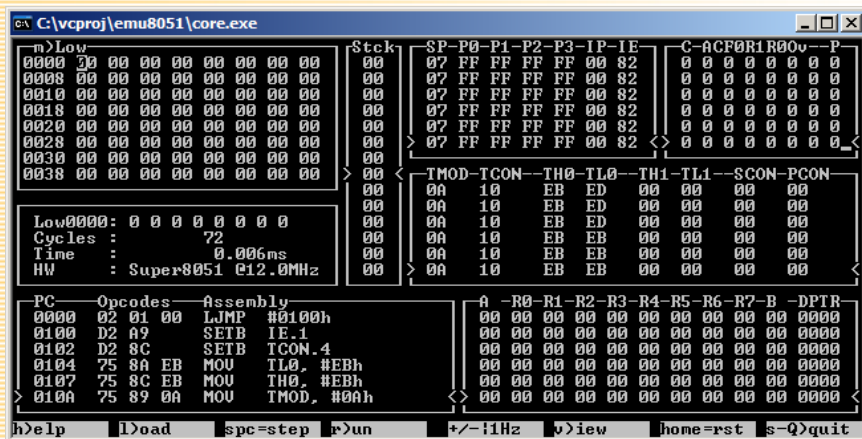
P0.0/1: 00h   P2.0/1: 00h
P0.2/3: 00h   P2.2/3: 00h
P0.4/5: 00h   P2.4/5: 00h
P0.6/7: 00h   P2.6/7: 00h

P1.0/1: 00h   P3.0/1: 00h
P1.2/3: 00h   P3.2/3: 00h
P1.4/5: 00h   P3.4/5: 00h
P1.6/7: 00h   P3.6/7: 00h

->X 8bit shift registers >

h>elp  l>oad  spc=step  r>un  +/-!Hz  v>iew  home=rst  s-Q>quit
```

- 'Logic board' stole the show



```
C:\vcproj\emu8051\core.exe
n>Low
0000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0008 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0018 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0028 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0038 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Stack
00 07 FF FF FF FF FF 00 82 00 00 00 00 00 00
08 07 FF FF FF FF FF 00 82 00 00 00 00 00 00
10 07 FF FF FF FF FF 00 82 00 00 00 00 00 00
18 07 FF FF FF FF FF 00 82 00 00 00 00 00 00
20 07 FF FF FF FF FF 00 82 00 00 00 00 00 00
28 07 FF FF FF FF FF 00 82 00 00 00 00 00 00
30 07 FF FF FF FF FF 00 82 00 00 00 00 00 00
38 07 FF FF FF FF FF 00 82 00 00 00 00 00 00

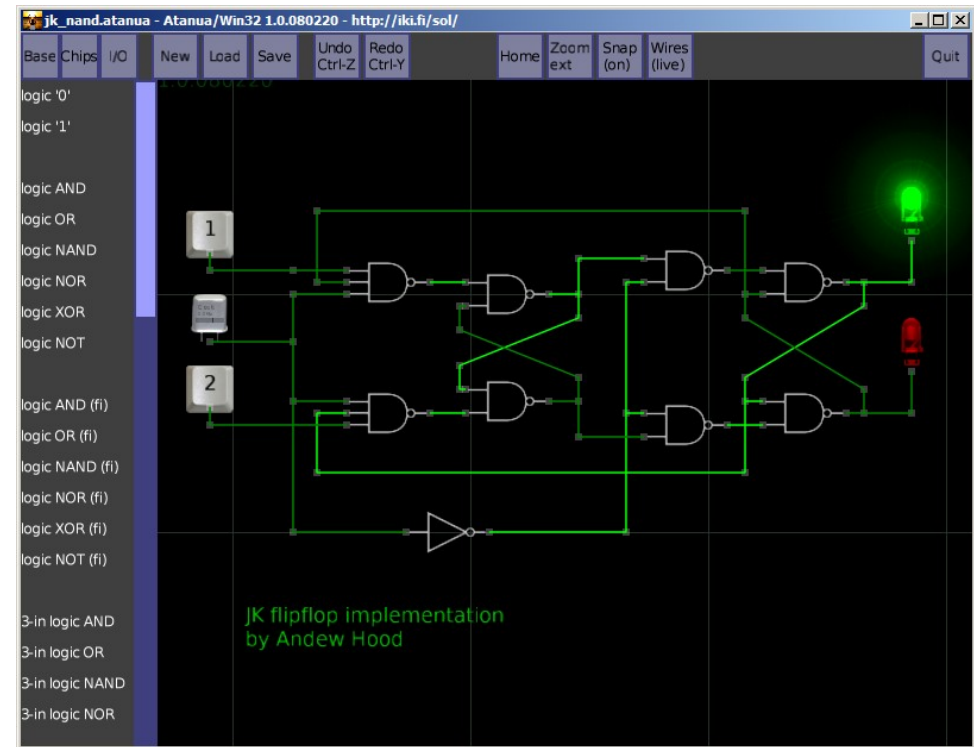
C-ACF0R1R00v--P
00 0A 10 EB ED 00 00 00 00
08 0A 10 EB ED 00 00 00 00
10 0A 10 EB ED 00 00 00 00
18 0A 10 EB ED 00 00 00 00
20 0A 10 EB ED 00 00 00 00
28 0A 10 EB ED 00 00 00 00
30 0A 10 EB ED 00 00 00 00
38 0A 10 EB ED 00 00 00 00

IMOD-TCON-TH0-TL0--TH1-TL1--SCON-PCON
00 00 00 00 00 00 00 00 00 00 00 00
08 00 00 00 00 00 00 00 00 00 00 00
10 00 00 00 00 00 00 00 00 00 00 00
18 00 00 00 00 00 00 00 00 00 00 00
20 00 00 00 00 00 00 00 00 00 00 00
28 00 00 00 00 00 00 00 00 00 00 00
30 00 00 00 00 00 00 00 00 00 00 00
38 00 00 00 00 00 00 00 00 00 00 00

Low0000: 0 0 0 0 0 0 0 0
Cycles : 72
Time : 0.006ms
HW : Super8051 @12.0MHz

PC Opcodes Assembly
0000 02 01 00 LJMPC #0100h
0100 D2 09 SETB IE.1
0102 D2 0C SETB TCON.4
0104 75 8A E8 MOV TL0, #E8h
0107 75 8C E8 MOV TH0, #E8h
010A 75 89 0A MOV TMOD, #0Ah

h>help l>load spc=step r>un +/-!Hz v>view home=rst s-Q>quit
```



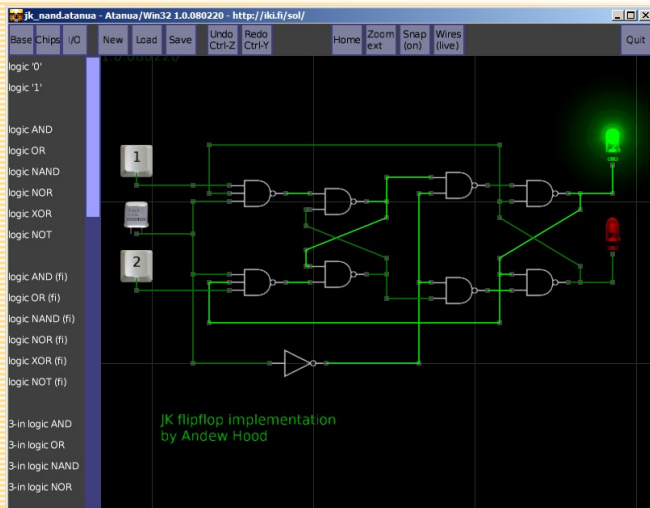


```
#include "atanua.h"
#include "atanua_internal.h"
#include "pluginchip.h"
#include "fileutils.h"

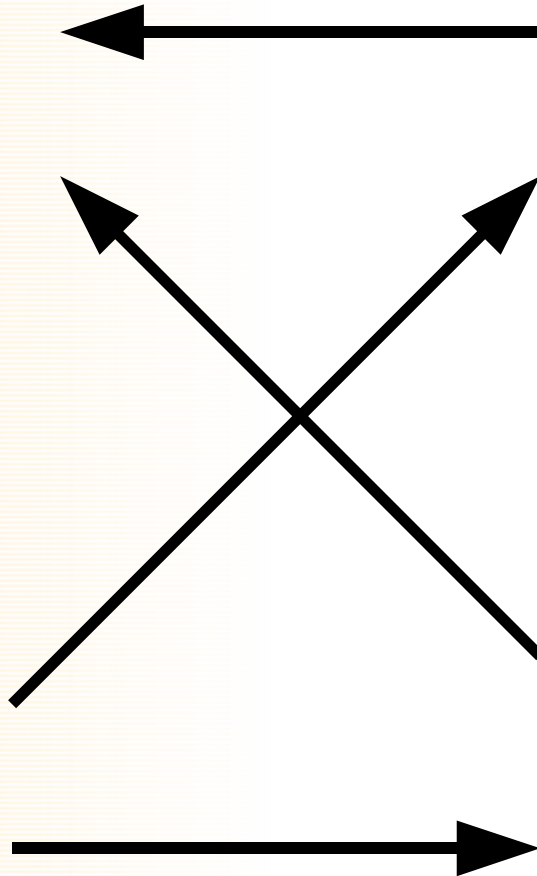
PluginChip::PluginChip(int aDllHandle, const char *aChipname)
{
    dlicreate = (createproc)getdllproc(aDllHandle, "create");
    dilupdate = (updateproc)getdllproc(aDllHandle, "update");
    dilrender = (renderproc)getdllproc(aDllHandle, "render");
    dilcleanup = (cleanupproc)getdllproc(aDllHandle, "cleanup");
    if (dlicreate == NULL || dilupdate == NULL || dilrender == NULL || dilcleanup == NULL)
        dlicreate = (mChipInfo.mPinCount == 0)
        {
            dlicreate = NULL;
            return;
        }
    set(0, 0, mChipInfo.mPinWidth, mChipInfo.mHeight, mChipInfo.mTooltips);
}

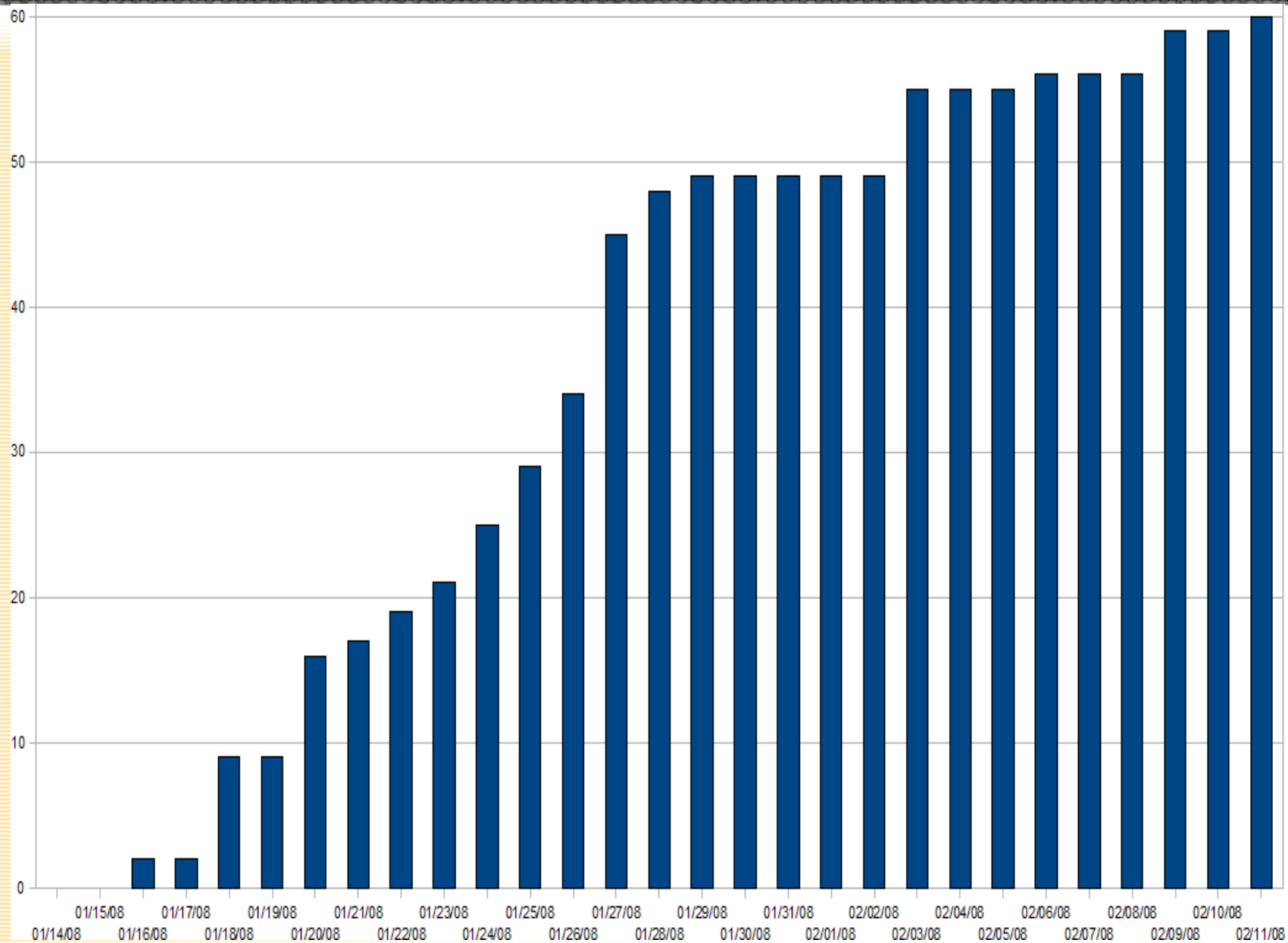
set(0, 0, mChipInfo.mPinWidth, mChipInfo.mHeight, mChipInfo.mTooltips);
int i;
for (i = 0; i < mChipInfo.mPinCount; i++)
{
    const char *tt = NULL;
    Pin *p = new Pin;
    mPin.push_back(p);
    if (mChipInfo.mPinTooltips)
        tt = mChipInfo.mPinTooltips[i];
```

```
[screen 0: bash] jarikom@shell:~$
Ludum Dare - http://www.imitationpickles.org/ludum/ - LD11 - April 18-20 - Yaay
18:25 < fydo> drZool: rock on
18:25 < drZool> yeah!
18:26 < drZool> never done something like this before, scary and relieving at
the same time
18:33 -!- mark [markf@c-71-231-111-167.hsd1.wa.comcast.net] has joined
#ludumdare
18:33 -!- mode/#ludumdare [+o mark] by X3
18:38 -!- hamumu [chatzilla@72-173-105-105.cust.wildblue.net] has joined
#ludumdare
18:38 -!- mode/#ludumdare [+o hamumu] by X3
19:04 < X-Out> drZool: I have
19:04 < X-Out> I ended up out of work :)
19:10 < LoneStranger> :(
19:26 < fydo> meow
19:54 -!- kohai [koaha567@gar13-3-82-240-80-106.fbx.proxad.net] has joined
#ludumdare
19:59 < Sol_HSA> say something funny, I'm going to take a screenshot of irc for
my presentation.. =)
20:01 < kohai> i love G.W. Bush !
20:02 < kohai> is it funny enough ? :)
20:04 < Falkoni> something funny!
[20:07] [Sol_HSA(+1)] [4:#ludumdare(+nst)] [Act: 3.5]
[#ludumdare]
```



Subject	Started by	Replies	Views	Last post
Welcome to Atanua discussion	Sol_HSA	0	123	January 24, 2008, 10:20:12 PM
Would you pay?	Sol_HSA	2	75	February 29, 2008, 06:17:59 PM
Circuit Duplicator	WilliamSharkey	3	63	February 28, 2008, 10:37:40 PM
VELLERMAN KB055 Plugin	WilliamSharkey	8	70	February 26, 2008, 10:32:48 PM
Feature Request ( Increment Char of Cloned Buttons)	WilliamSharkey	0	18	February 26, 2008, 09:24:48 PM
Feature Request (Color Pallet)	WilliamSharkey	0	16	February 26, 2008, 05:42:41 PM
Known bugs..	Sol_HSA	0	22	February 25, 2008, 02:21:34 PM
Audio chip	Sol_HSA	3	63	February 23, 2008, 10:19:02 AM
Feature Requests	moe	14	212	February 21, 2008, 11:43:09 PM
Easter eggs..	Sol_HSA	0	47	February 21, 2008, 12:32:31 PM
Demo files	benjam	2	112	February 21, 2008, 02:44:41 AM
sol.Gfxile.net and Atanua are hard for me to remember	WilliamSharkey	1	57	February 20, 2008, 07:47:19 AM
Saved file jumbled	andrew77	4	97	February 10, 2008, 10:33:49 PM
Bug?	polakko	1	75	February 10, 2008, 08:34:09 AM
web-based flash demo	Sol_HSA	0	58	February 10, 2008, 08:34:09 AM
Reminds me of Robot Odyssey	andrew77	5	193	February 05, 2008, 05:09:31 PM





Simulated components

jk\_nand.atanua - Atanua/Win32 1.0.080220 - <http://iki.fi/sol/>

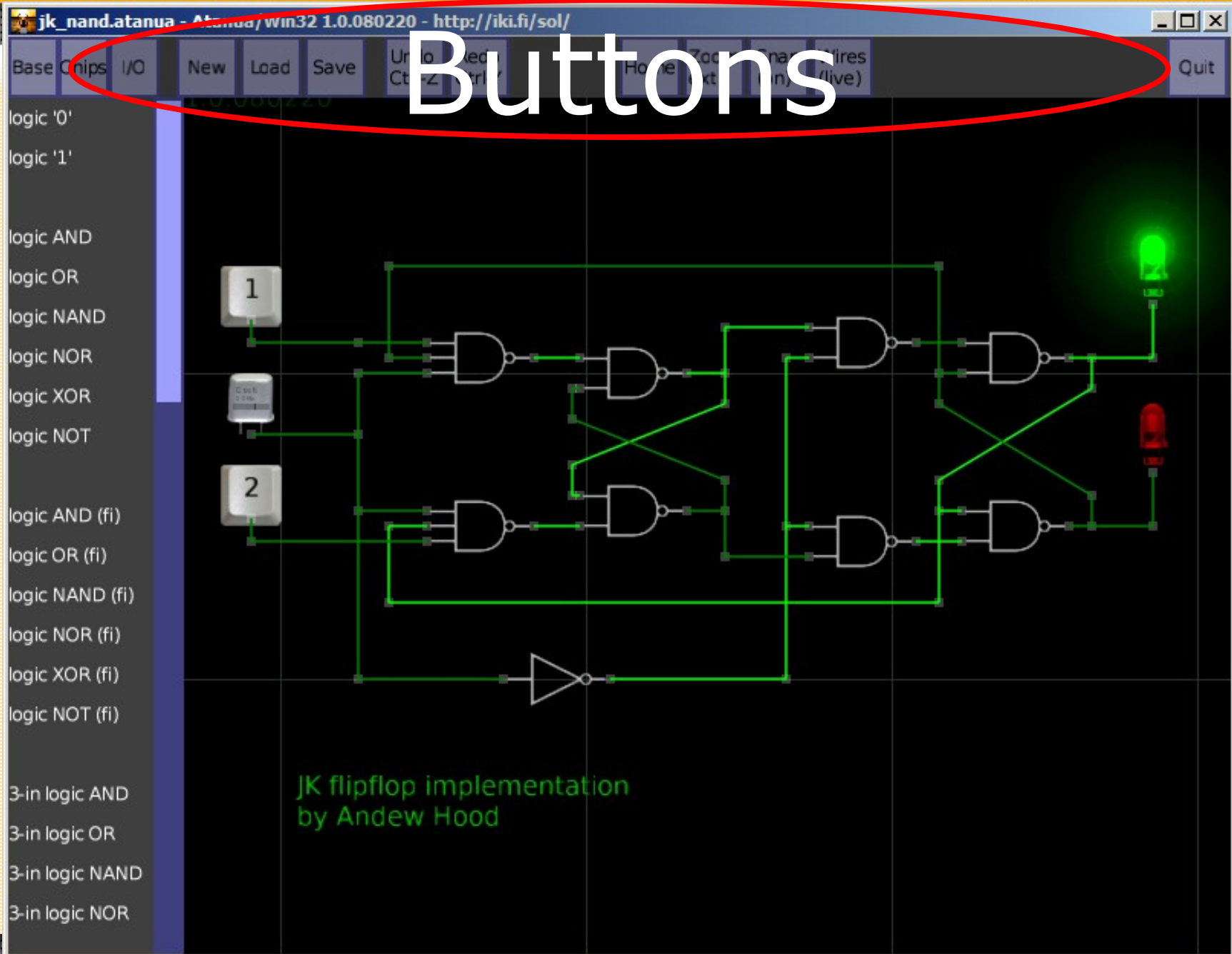
Base Chips I/O New Load Save Undo Ctrl-Z Redo Ctrl-Y Home Zoom ext Snap (on) Wires (live) Quit

logic '0'  
logic '1'  
logic AND  
logic OR  
logic NAND  
logic NOR  
logic XOR  
logic NOT  
logic AND (fi)  
logic OR (fi)  
logic NAND (fi)  
logic NOR (fi)  
logic XOR (fi)  
logic NOT (fi)  
3-in logic AND  
3-in logic OR  
3-in logic NAND  
3-in logic NOR

JK flipflop implementation  
by Andrew Hood



# Buttons



The screenshot shows the Atanua logic simulator interface. The title bar reads "jk\_nand.atanua - Atanua/Win32 1.0.080220 - http://iki.fi/sol/". The menu bar includes "Base Chips", "VO", "New", "Load", "Save", "Undo Ctrl-Z", "Redo Ctrl-Y", "Home", "Zoom ext", "Snap (on)", "Wires (live)", and "Quit". A red oval highlights the "Base Chips" menu, which lists various logic components such as "logic '0'", "logic '1'", "logic AND", "logic OR", "logic NAND", "logic NOR", "logic XOR", "logic NOT", and various multi-input logic gates. The main workspace displays a circuit diagram of a JK flipflop implemented using NAND gates. The circuit includes a constant logic '1' input, a clock input, and two outputs: a green LED (labeled "LED1") and a red LED (labeled "LED2"). The text "JK flipflop implementation by Andrew Hood" is visible at the bottom of the workspace.

# Toolkit

jk\_nand.atanua - Atanua/Win32 1.0.080220 - http://iki.fi/sol/

Base Chips I/O New Load Save Undo Ctrl-Z Redo Ctrl-Y Home Zoom ext Snap (on) Wires (live) Quit

logic '0'  
logic '1'  
logic AND  
logic OR  
logic NAND  
logic NOR  
logic XOR  
logic NOT  
logic AND (fi)  
logic OR (fi)  
logic NAND (fi)  
logic NOR (fi)  
logic XOR (fi)  
logic NOT (fi)  
3-in logic AND  
3-in logic OR  
3-in logic NAND  
3-in logic NOR

JK flipflop implementation  
by Andrew Hood

Work area

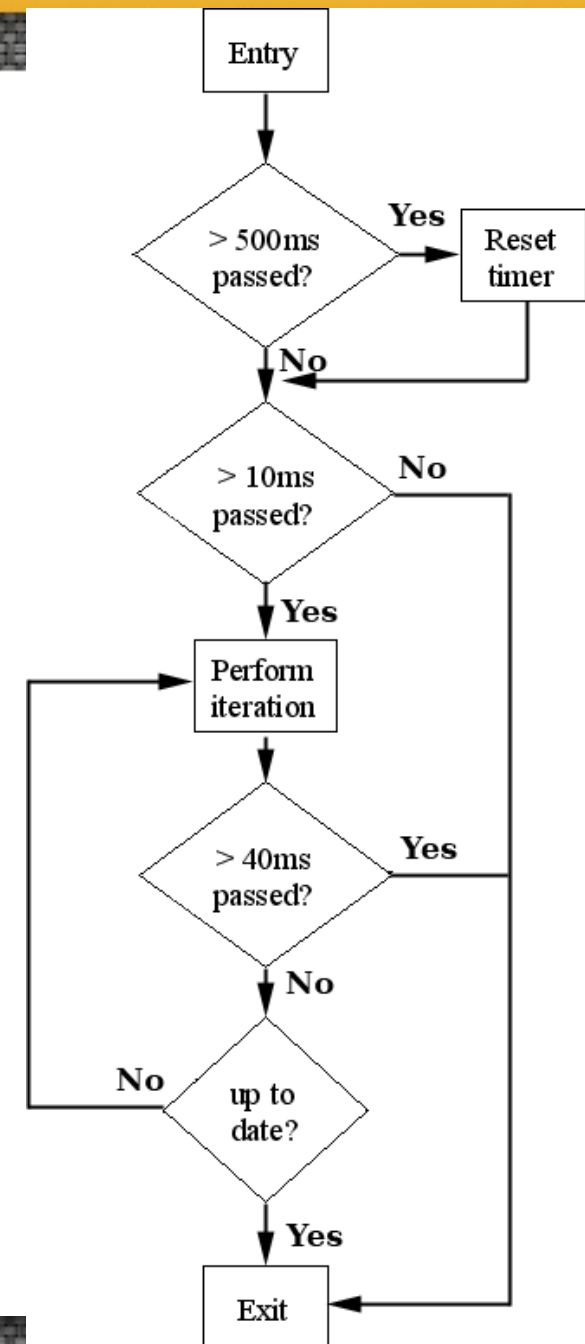
Demo..

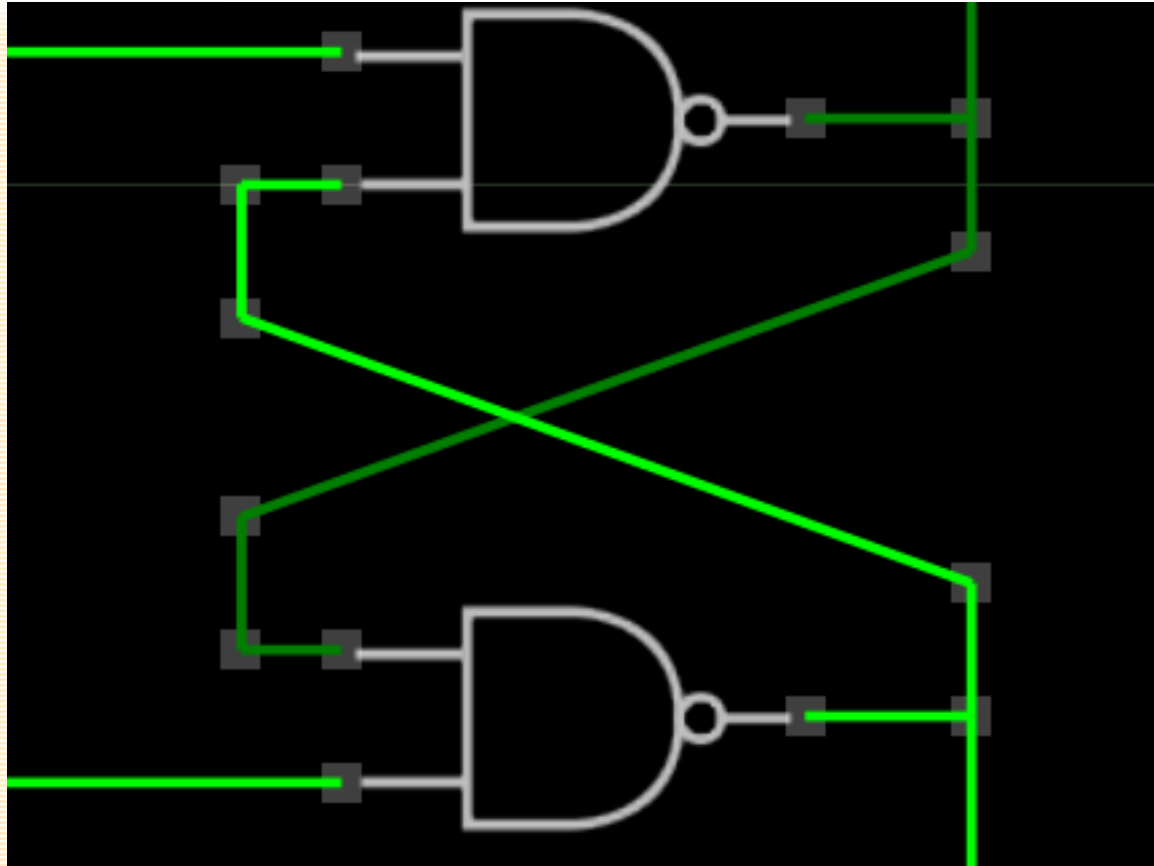


# Questions?

- Atanua: <http://www.atanua.org>
- My site: <http://iki.fi/sol/>
- 8051 emu: <http://iki.fi/sol/8051.html>

- Back-up slide:  
Implementation of the  
real-time logic simulation
- Logic simulation loop runs  
at 1000Hz regardless of  
graphics frame rate.





- Back-up slide:  
Problematic cross-connections