



Universidad de Costa Rica

FACULTAD DE INGENIERÍA ELÉCTRICA

TAREA 1

Autor:
Luis Daniel Ferreto Chavarría

Carné: B82958

August 2021

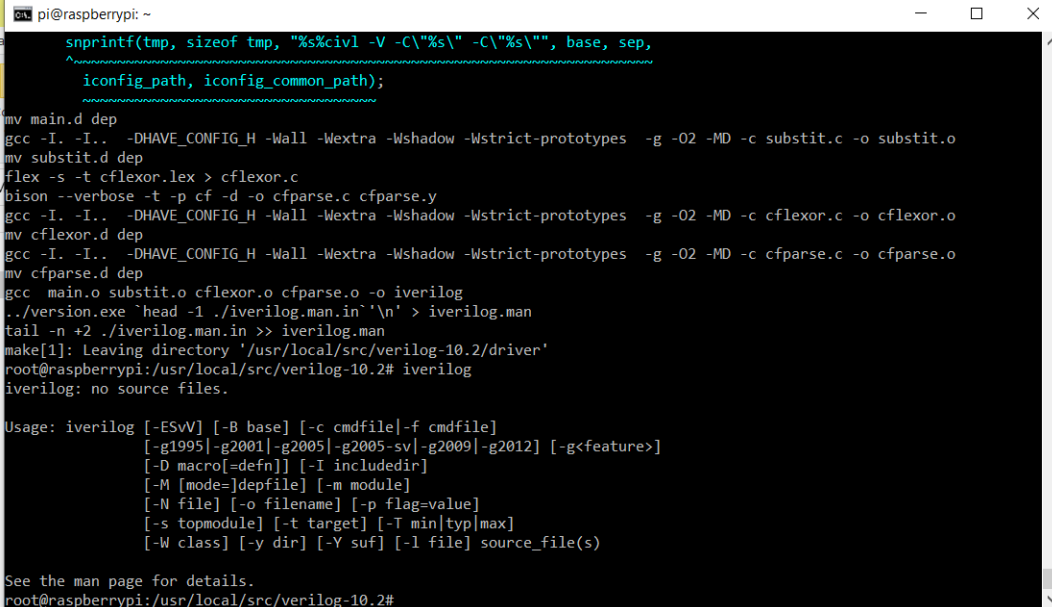
0.1. Contabilidad del tiempo invertido

- Compresión del enunciado e instrucciones de la tarea 5 minutos.
- Conexión a un servidor Linux (Raspberry pi) mediante ssh 1 minuto.
- Instalación de los programas a utilizar mediante consola en GNU/Linux 6 minutos (dependiente de la velocidad de internet).
- Comprensión de las herramientas 15 minutos.
- Creación del archivo Makefile para cada una de las instrucciones de la tarea 5 minutos.
- Comando seed: La documentación y creación de la línea de ejecución 10 minutos.
- Comando seed: La documentación y creación de la línea de ejecución 10 minutos. Comando autoinst: La documentación y creación de la línea de ejecución 15 minutos.

0.2. Instalación Herramientas

La instalación y uso de herramientas, se realizó en Linux en la distribución Raspbian.

Las herramientas instaladas son; iverilog, GTKWave y Yosys para instalar cada una, se utilizó el comando en terminal de linux (con root) **apt-get install <PAQUETE>** y para verificar su instalación, se ingresó en la terminal el nombre del paquete y de manera opcional el versionamiento del mismo, a continuación, se muestra la demostración de los 3 programas.



```

pi@raspberrypi: ~
snprintf(tmp, sizeof tmp, "%s%civl -V -C \"%s\" -C \"%s\"", base, sep,
^~~~~~
iconfig_path, iconfig_common_path);
~~~~~
mv main.d dep
gcc -I. -I.. -DHAVE_CONFIG_H -Wall -Wextra -Wshadow -Wstrict-prototypes -g -O2 -MD -c substit.c -o substit.o
mv substit.d dep
flex -s -t cflexor.lex > cflexor.c
bison --verbose -t -p cf -d -o cfpase.c cfpase.y
gcc -I. -I.. -DHAVE_CONFIG_H -Wall -Wextra -Wshadow -Wstrict-prototypes -g -O2 -MD -c cflexor.c -o cflexor.o
mv cflexor.d dep
gcc -I. -I.. -DHAVE_CONFIG_H -Wall -Wextra -Wshadow -Wstrict-prototypes -g -O2 -MD -c cfpase.c -o cfpase.o
mv cfpase.d dep
gcc main.o substit.o cflexor.o cfpase.o -o iverilog
./version.exe `head -1 ./iverilog.man.in` > iverilog.man
tail -n +2 ./iverilog.man.in >> iverilog.man
make[1]: Leaving directory '/usr/local/src/verilog-10.2/driver'
root@raspberrypi:/usr/local/src/verilog-10.2# iverilog
iverilog: no source files.

Usage: iverilog [-ESvW] [-B base] [-c cmdfile] [-f cmdfile]
      [-g1995] [-g2001] [-g2005] [-g2005-sv] [-g2009] [-g2012] [-g<feature>]
      [-D macro[=defn]] [-I includedir]
      [-M [mode=]depfile] [-m module]
      [-N file] [-o filename] [-p flag=value]
      [-s topmodule] [-t target] [-T min|typ|max]
      [-W class] [-y dir] [-Y suf] [-l file] source_file(s)

See the man page for details.
root@raspberrypi:/usr/local/src/verilog-10.2#

```

Figura 1: Visualización de la información del programa iverilog

```

pi@raspberrypi: ~
root@raspberrypi:/usr/local/src/verilog-10.2# root@raspberrypi:/usr/local/src/verilog-10.2# gtkwave --v
could not initialize GTK! Is DISPLAY env var/xhost set?

Usage: gtkwave [OPTION]... [DUMPFIL] [SAVEFILE] [RCFILE]

-n, --nocli=DIRPATH      use file requester for dumpfile name
-f, --dump=FILE           specify dumpfile name
-F, --fastload           generate/use VCD recoder fastload files
-o, --optimize           optimize VCD to FST
-a, --save=FILE          specify savefile name
-A, --autosavename       assume savefile is suffix modified dumpfile name
-r, --rcfile=FILE        specify override .rcfile name
-d, --defaultskip        if missing .rcfile, do not use useful defaults
-D, --dualid=WHICH       specify multisession identifier
-l, --logfile=FILE       specify simulation logfile name for time values
-s, --start=TIME         specify start time for LXT2/VZT block skip
-e, --end=TIME           specify end time for LXT2/VZT block skip
-t, --stems=FILE         specify stems file for source code annotation
-c, --cpu=NUMCPUS        specify number of CPUs for parallelizable ops
-N, --nowm              disable window manager for most windows
-M, --nomenus           do not render menubar (for making applets)
-S, --script=FILE        specify Tcl command script file for execution
-T, --tcl_init=FILE      specify Tcl command script file to be loaded on startup
-W, --wish              enable Tcl command line on stdio
-R, --repscript=FILE     specify timer-driven Tcl command script file
-P, --repperiod=VALUE    specify repscript period in msec (default: 500)
-X, --xid=XID            specify XID of window for GtkPlug to connect to
-1, --rpcid=RPCID        specify RPCID of GConf session
-2, --chdir=DIR          specify new current working directory
-3, --restore            restore previous session

```

Figura 2: Visualización de la información del programa GTKWave

```

pi@raspberrypi: ~
root@raspberrypi:/usr/local/src/verilog-10.2# root@raspberrypi:/usr/local/src/verilog-10.2# yosys --v
yosys: invalid option -- '-'
Run 'yosys -h' for help.
root@raspberrypi:/usr/local/src/verilog-10.2# yosys

/-----/
|
|  yosys -- Yosys Open SYnthesis Suite
|
|  Copyright (C) 2012 - 2018 Clifford Wolf <clifford@clifford.at>
|
|  Permission to use, copy, modify, and/or distribute this software for any
|  purpose with or without fee is hereby granted, provided that the above
|  copyright notice and this permission notice appear in all copies.
|
|  THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES
|  WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF
|  MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR
|  ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES
|  WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN
|  ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF
|  OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.
|
|-----/

```

Figura 3: Visualización de la información del programa Yosys

0.3. Descripción de herramientas

0.3.1. Iverlog

Esta herramienta es Icarus verilog que corresponde a un compilador de código de verilog de extensión .v, el cual cuenta con plugins en IDEs para compilar código directamente.

Para ejecutar un programa .v con este compilado, primeramente se necesita generar un archivo objeto el cual al ejecutarse con vvp puede dar como salida, archivo vcd para GTKWave o una salida en la terminal, a continuación con el primer target del Makefile la salida que genera este compilador en terminal.

```

pi@raspberrypi: ~/Desktop/digitales/digitales2/Tarea1
iverilog -o probador probador.v
pi@raspberrypi:~/Desktop/digitales/digitales2/Tarea1 $ make p2
vvp probador
VCD info: dumpfile alarma.vcd opened for output.
      clk,      sAlr_estr,      sAlr_cond,      sLuz,      sPrta,      sIgn
      0 0      z      z      0 0      0
      2 1      z      z      0 0      1
      4 0      z      z      0 0      1
      6 1      z      z      0 1      0
      8 0      z      z      0 1      0
     10 1      z      z      0 1      1
     12 0      z      z      0 1      1
     14 1      z      z      1 0      0
     16 0      z      z      1 0      0
     18 1      z      z      1 0      1
     20 0      z      z      1 0      1
     22 1      z      z      1 1      0
     24 0      z      z      1 1      0
     26 1      z      z      1 1      1
     28 0      z      z      1 1      1
     30 1      z      z      0 0      0
     32 0      z      z      0 0      0
     34 1      z      z      0 0      0

pi@raspberrypi:~/Desktop/digitales/digitales2/Tarea1 $ la
-bash: la: command not found
pi@raspberrypi:~/Desktop/digitales/digitales2/Tarea1 $ ls
2021-08-30-130736_1280x720_screenshot.png  alarma_desc_estructural.v  BancoPrueba.o  probador
alarma_desc_conductual.v                  alarma.vcd                BancoPrueba.v  probador.v
alarma_desc_conductual.v~                  BancoPruebaConductual.v  Makefile       windows_script.bat
pi@raspberrypi:~/Desktop/digitales/digitales2/Tarea1 $

```

Figura 4: Funcionalidad de Icarus verilog

como se muestra en la figura 4, se muestra la salida en terminal y se genera el archivo .vcd.

0.3.2. GTKWave

La herramienta consiste en un entorno gráfico para mostrar las señales generadas al compilar un archivo .v, cabe destacar que se puede configurar el tiempo de la señal, el tamaño, entre otros parámetros.

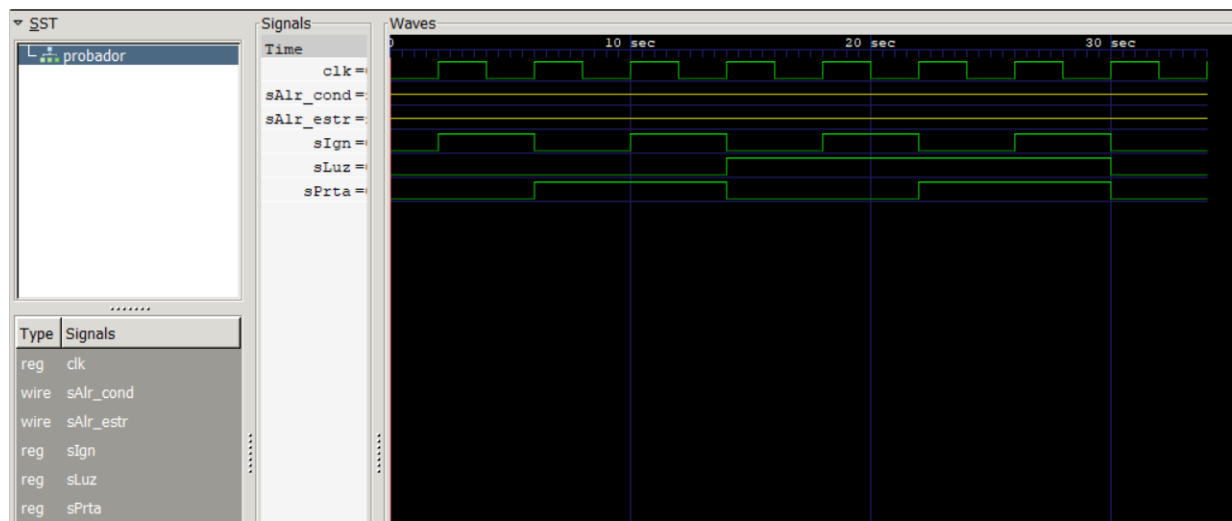


Figura 5: Funcionalidad de GTKWave

0.3.3. Yosys

La herramienta Yosys permite obtener un diagrama de bloques a la salida donde este lee un archivo .v y para mostrar esto como diagrama de bloques, se utiliza para compilar "show".

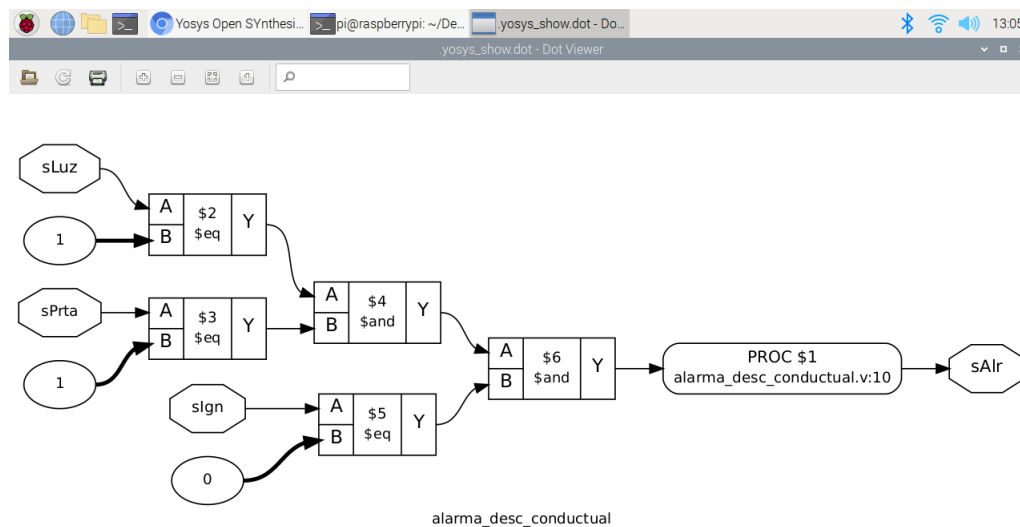


Figura 6: Funcionalidad de Yosys

0.4. Makefile

Se creó un archivo Makefile para automatizar la compilación del código y automatizar las tareas, el código del Makefile cuenta con 5 instrucciones o targets, donde el primero de ellos es para generar un archivo con la compilación del código fuente de verilog, la segunda es para ejecutar el archivo, la tercera es para visualizar en GTKWave las señales en el dominio del tiempo, la cuarta para la utilización de Yosys, la quinta para el sed y la última para el AUTOINST con emacs.

Para ejecutar todo el Archivo Makefile, se utiliza en la terminal bash **make** y si es solo un target en específico **make <target>**

```

Makefile
C: > Users > Daniel > Box > DigitalesII > DigitalesII > Mis tarea > Tarea1 > Makefile
1 all: programa p2 p3 p4 p5
2 programa:
3     iverilog -o probador probador.v
4 p2:
5     vvp probador
6 p3:
7     gtkwave alarma.vcd
8 p4:
9     yosys -p "read -sv alarma_desc_conductual.v" -p "hierarchy -top alarma_desc_conductual" -p "show"
10 p5:
11     sed 's/alarma_desc_conductual/alarma_desc_conductual_synth/' alarma_desc_conductual.v
12 p6:
13     emacs -batch autoinst.v(/(*AUTOINST*/));
14 clean:
15     rm -r *.vcd

```

Figura 7: Visualización del código Makefile

0.5. Sed

Para realizar el sed, se copió el archivo alarma_desc_conductual.v a otro archivo alarma_desc_conductualc.v y por medio de la instrucción en el Makefile, se le cambió el nombre al módulo y por medio del target p5 del Makefile, se generó el sed.

```

pi@raspberrypi: ~/Desktop/digitales/digitales2/Tarea1 $ ls
2021-08-30-130736_1280x720_screenshot.png  BancoPrueba.o
alarm_desc_conductual.v                    BancoPrueba.v
alarm_desc_conductual.v~                   Makefile
alarm_desc_estructural.v                  probador.v
alarm_desc_estructural.v~                  windows_script.bat
BancoPruebaConductual.v                   alarm_desc_conductual.v
pi@raspberrypi: ~/Desktop/digitales/digitales2/Tarea1 $ cp alarm_desc_conductual.v alarm_desc_conductualc.v
pi@raspberrypi: ~/Desktop/digitales/digitales2/Tarea1 $ make p5
sed 's/alarm_desc_conductual/alarm_desc_conductual_synth/' alarm_desc_conductualc.v
module alarm_desc_conductual_synth (
    output reg    sAlr, // sAlr de tipo reg, almacena el valor
    input         sluz, // No se indica el tipo de sluz, wire implícito
    input         sPrta, // No se indica el tipo de sPrta, wire implícito
    input         sIgn); // No se indica el tipo de sIgn, wire implícito
    // En las descripciones conductuales, los puertos de entrada son wires.
    // Los puertos de salida pueden ser wires o regs, dependiendo de la
    // implementación.

    always @ (*) begin // always combinacional, bloque de procedimiento/comportamiento
        // (*) Lista de sensibilidad, entra al "always" ante cualquier cambio
        // en (sluz or sPrta or sIgn)
        if (sluz == 1 & sPrta == 1 & sIgn == 0)
            sAlr = 1; // Asignación bloqueante (=)
        else
            sAlr = 0; // Asignación bloqueante (=)
        end
    endmodule
pi@raspberrypi: ~/Desktop/digitales/digitales2/Tarea1 $

```

Figura 8: Generación del Seed

0.6. AUTOINST

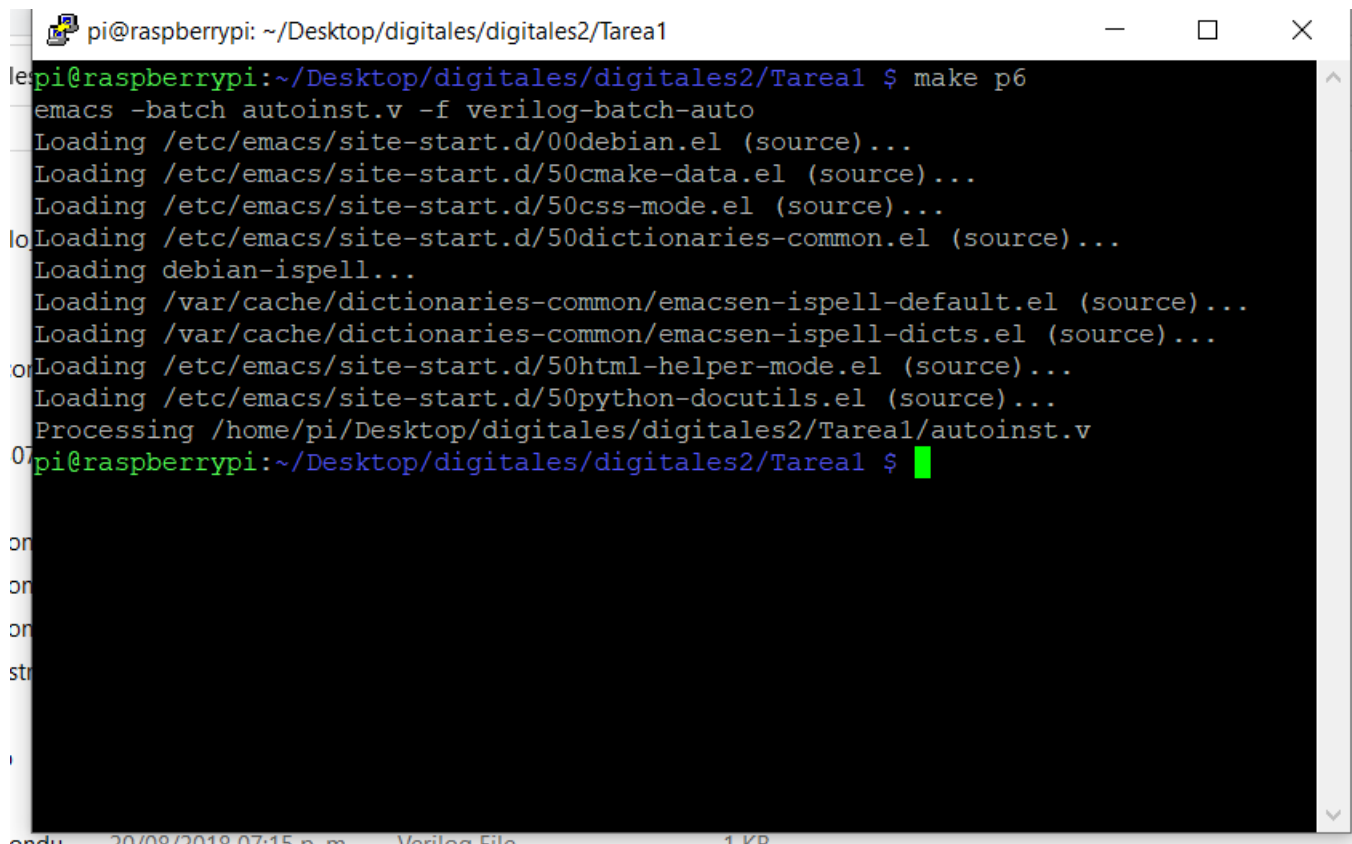
Para hacer el AUTOINST, se descargó emacs en Linux en consola. Posteriormente, como se observa en la figura 9 se creó un archivo autoinst.v el cual es un módulo de instancia. En este se incluyó el archivo alarma desconductual.v y mediante el Makefile se ejecutó el autoinst y la salida se muestra en 11

```

C: > Users > Daniel > Box > DigitalesII > DigitalesII > Mis tarea > Tarea1 > autoinst.v
1 | include "alarm_desc_conductual.v"
2
3 | module Newmodule;
4 |
5 |     alarm_desc_conductual alarm_desc_conductual(/*AUTOINST*/);
6 | endmodule
7
8
9

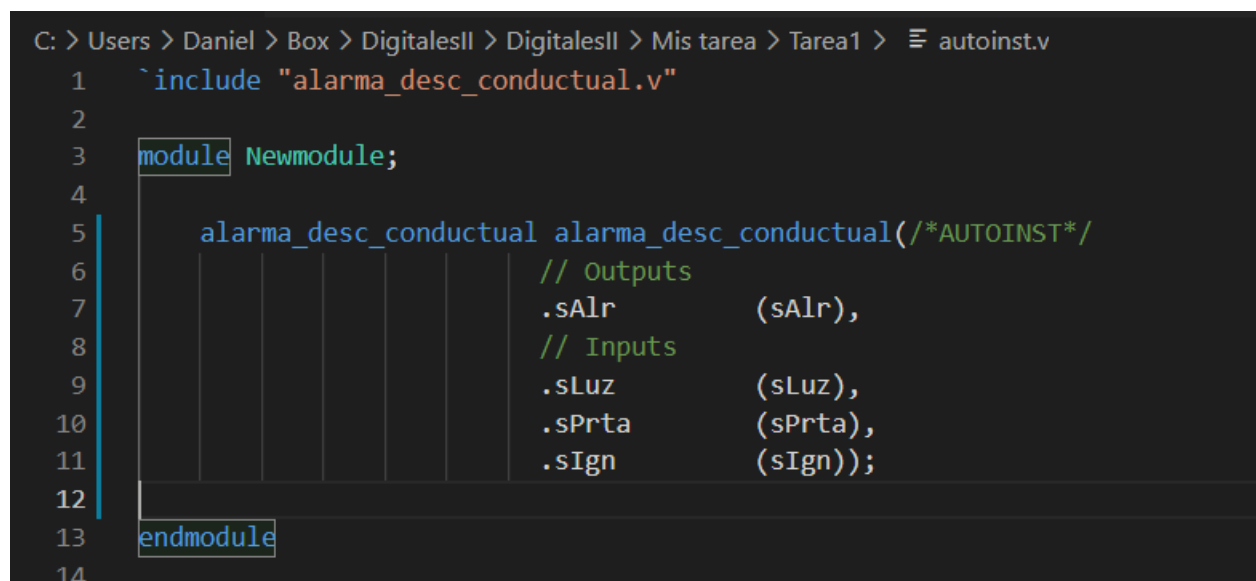
```

Figura 9: Código para AUTOINST



```
pi@raspberrypi: ~/Desktop/digitales/digitales2/Tarea1
pi@raspberrypi:~/Desktop/digitales/digitales2/Tarea1 $ make p6
emacs -batch autoinst.v -f verilog-batch-auto
Loading /etc/emacs/site-start.d/00debian.el (source)...
Loading /etc/emacs/site-start.d/50cmake-data.el (source)...
Loading /etc/emacs/site-start.d/50css-mode.el (source)...
Loading /etc/emacs/site-start.d/50dictionaries-common.el (source)...
Loading debian-ispell...
Loading /var/cache/dictionaries-common/emacs-en-ispell-default.el (source)...
Loading /var/cache/dictionaries-common/emacs-en-ispell-dicts.el (source)...
Loading /etc/emacs/site-start.d/50html-helper-mode.el (source)...
Loading /etc/emacs/site-start.d/50python-docutils.el (source)...
Processing /home/pi/Desktop/digitales/digitales2/Tarea1/autoinst.v
pi@raspberrypi:~/Desktop/digitales/digitales2/Tarea1 $
```

Figura 10: Ejecución de el AUTOINST en terminal



```
C: > Users > Daniel > Box > DigitalesII > DigitalesII > Mis tarea > Tarea1 > autoinst.v
1  `include "alarma_desc_conductual.v"
2
3  module Newmodule;
4
5      alarma_desc_conductual alarma_desc_conductual(/*AUTOINST*/
6          // Outputs
7          .sAlr      (sAlr),
8          // Inputs
9          .sLuz      (sLuz),
10         .sPrta     (sPrta),
11         .sIgn      (sIgn));
12
13  endmodule
14
```

Figura 11: Salida de AUTOINST emacs

Bibliografía

- [1] Linuxito. (2018). *Primeros pasos con Icarus Verilog en GNU/Linux*. Recuperado el 30 de agosto de 2021. Desde, <https://www.linuxito.com/programacion/1091-primeros-pasos-con-icarus-verilog-en-gnu-linux>
- [2] Wolf, C., Glaser, J. (2013). Yosys - A Free Verilog Synthesis Suite. Proceedings of the 21st Austrian Workshop on Micro-electronics (Austrochip).
- [3] Mckay, A. (2020). How to Use the sed Command on Linux. Recuperado el 30 de agosto de 2021. Desde, <https://www.howtogeek.com/666395/how-to-use-the-sed-command-on-linux/>
- [4] Anónimo. (2019). Introduction of Verilog Mode and Emacs. Recuperado el 30 de agosto de 2021. Desde, [https://s311354.github.io/Louis.github.io/2019/02/24/Intorduce \$Verilog_{Mode}\$ and \$Emacs\$ /](https://s311354.github.io/Louis.github.io/2019/02/24/Intorduce$Verilog_{Mode}$and$Emacs$/)