Digital Smart Safe - EECS 3201 Final Project Report

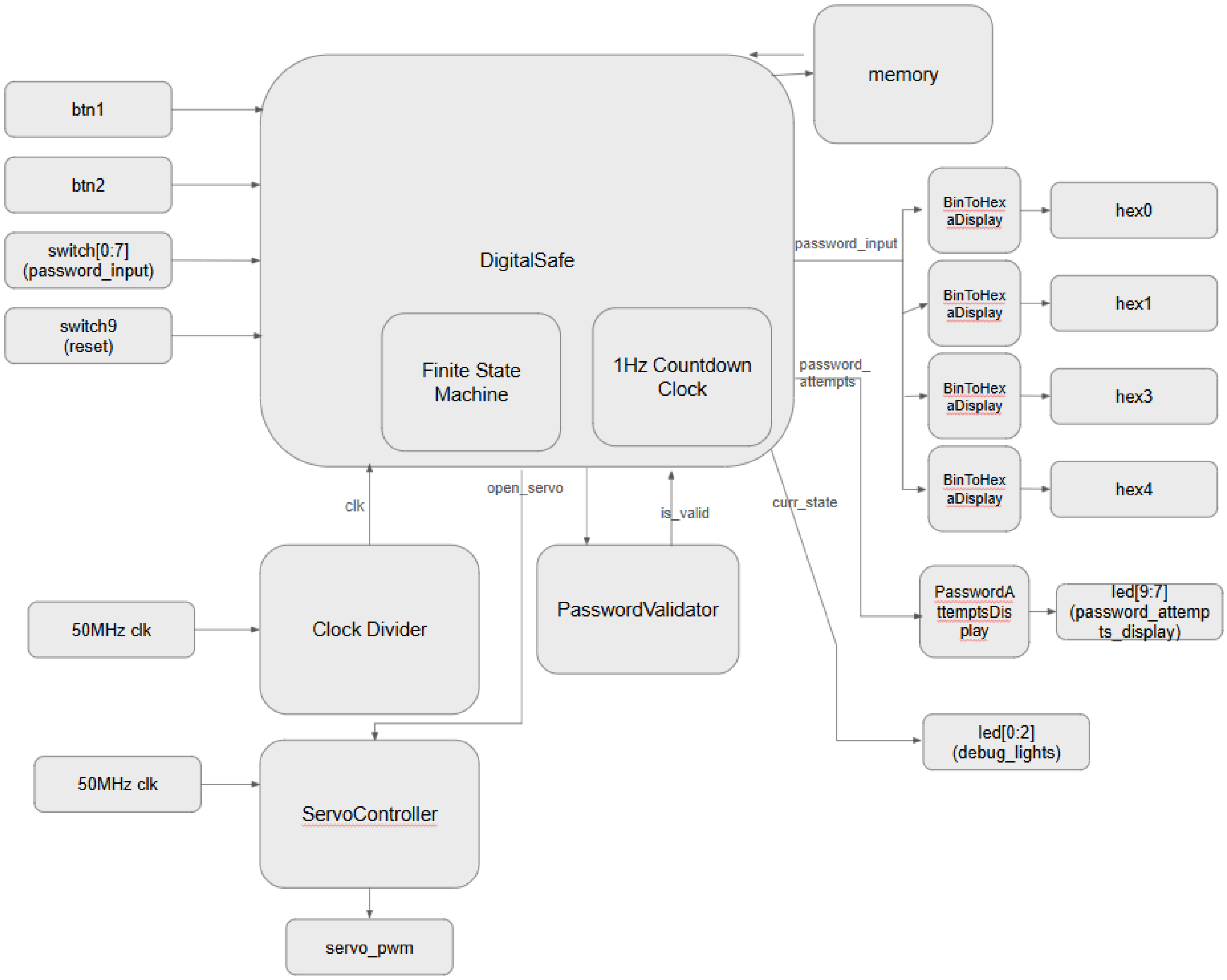
Ekaterina Kozlovsky
04/01/2025

Specification

This project implements a digital locking system using Quartus Prime Lite to program the DE10-Lite FPGA development board, a servo motor, and a switch based binary passcode mechanism. The goal is to simulate a secure digital safe where users can set, enter, and change a password. The system implements a user interface via switches, LEDs, and 7 segment displays to provide real time feedback. A state machine manages and controls the different functions of the system, including input handling, password verification, lockout, and servo control. The servo motor physically locks or unlocks the system, enhancing the realism of the digital safe. After three incorrect attempts, a three minute lockout countdown is triggered, blocking further input until a reset timer completes.

Architecture

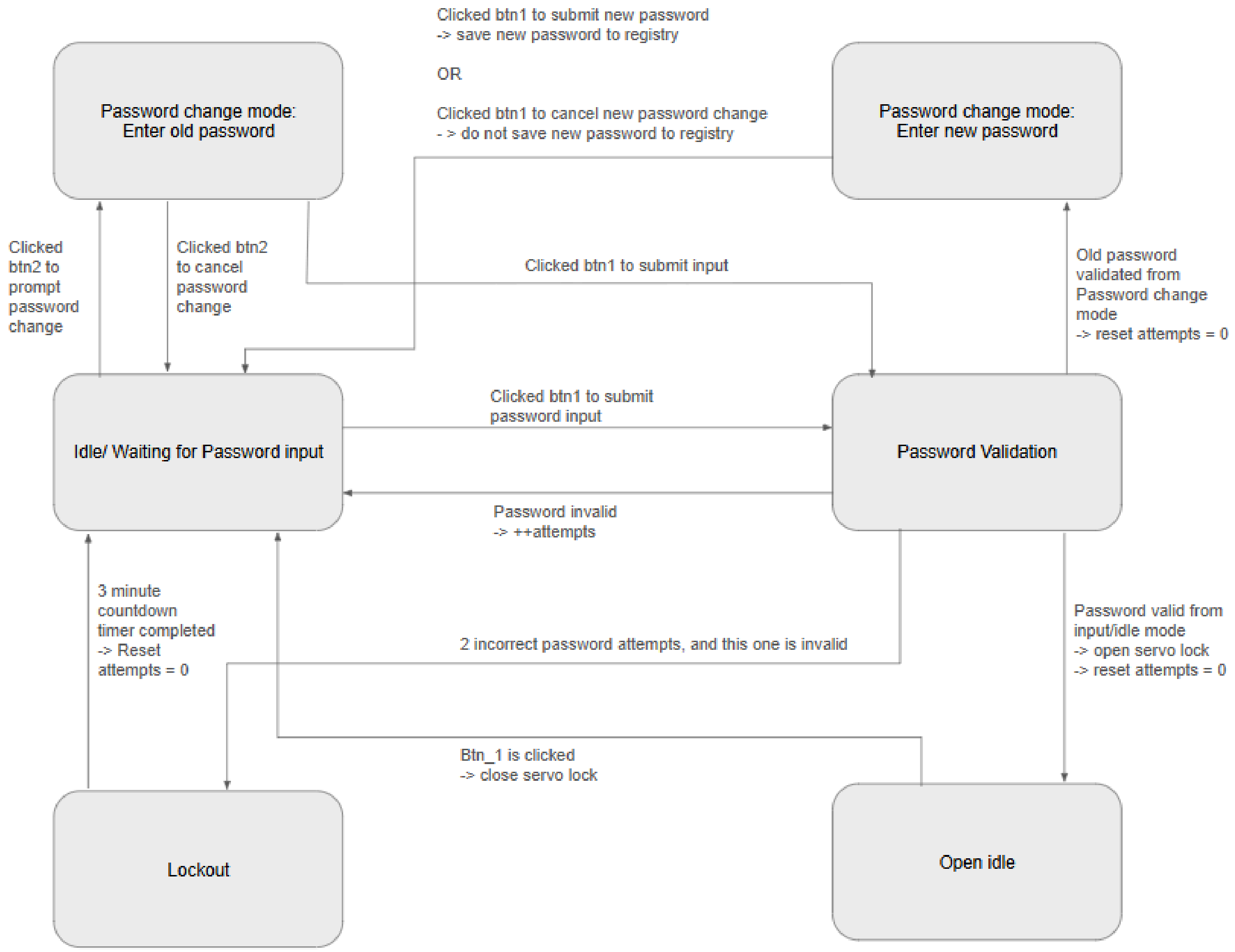
The Digital Smart Safe is built on a DE10-Lite board and uses a finite state machine (FSM) with integrated I/O control, password management, and hardware components such as servo motors, LED lights, and 7 segment displays. The system supports user input through the use of switches and buttons on the board.



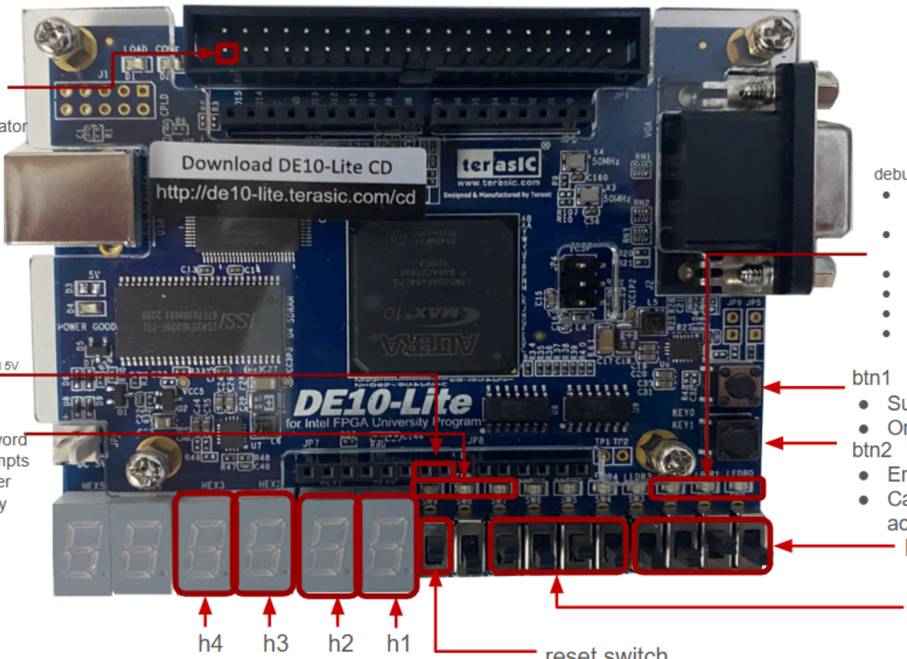
The FSM has 6 primary states:

- IDLE: Waits for user action.
- PASSWORD_VALIDATION: Verifies the entered password.
- PASSWORDCHANGE_OLDPASS: User needs to enter their current password before updating it.

  PASSWORDCHANGE_NEWPASS: Accepts a new password.
- LOCKOUT: Activates after three failed attempts, with a countdown timer.
- OPEN_IDLE: Opens servo, and displays "OPEN" on the 7 segment display.


Password change mode:
Enter old password
Password change mode:
Enter new password
Clicked btn1 to submit new password
-> save new password to registry
OR
Clicked btn1 to cancel new password change
- > do not save new password to registry
Clicked btn2 to prompt password change
Clicked btn2 to cancel password change
Clicked btn1 to submit input
Old password validated from Password change mode
-> reset attempts = 0
Idle/ Waiting for Password input
Clicked btn1 to submit password input
Password Validation
Password invalid
-> ++attempts
3 minute countdown timer completed
-> Reset attempts = 0
2 incorrect password attempts, and this one is invalid
Password valid from input/idle mode
-> open servo lock
-> reset attempts = 0
Btn_1 is clicked
-> close servo lock
Lockout
Open idle

Module	Function
DigitalSafe (Top-level module)	Manages the state transitions. Responds to button inputs and triggers changes in password logic, countdowns, or servo position.
PasswordValidator	Compares the entered password against the saved password using XNORs. Returns is_valid as an output to control state transition.
PasswordAttempts	Displays the number of failed password attempts using 3 bit output to LEDs. User feedback.
ClockDivider	Scales down the main DE10-Lite clock (50 MHz) to 100 ms.
ServoController	Generates PWM signals to open or close a servo motor (1.5ms or 2.5ms). Operates on a separate clock for stability.
BinToHexaDisplay	Converts 5-bit binary input to 7 segment output (plus decimal point bit). Used across the system for displaying password digits and countdown timers.



Servo PWM signal generator

GND and 5V for servo

password_attempts counter display

debug lights display current state

- 001: PASSWORDCHANGE_OLDPASS
- 010: PASSWORDCHANGE_NEWPASS
- 011: IDLE
- 100: PASSWORD_VALIDATION
- 101: LOCKOUT
- 110:OPEN_IDLE

btn1

- Submit input for validation
- Or save new password

btn2

- Enter change password mode
- Cancel password change action

password_input (second digit)

password_input (first digit)

h4 h3 h2 h1

- Displays user input
- Or lockout countdown
- Or 'Open'

reset switch

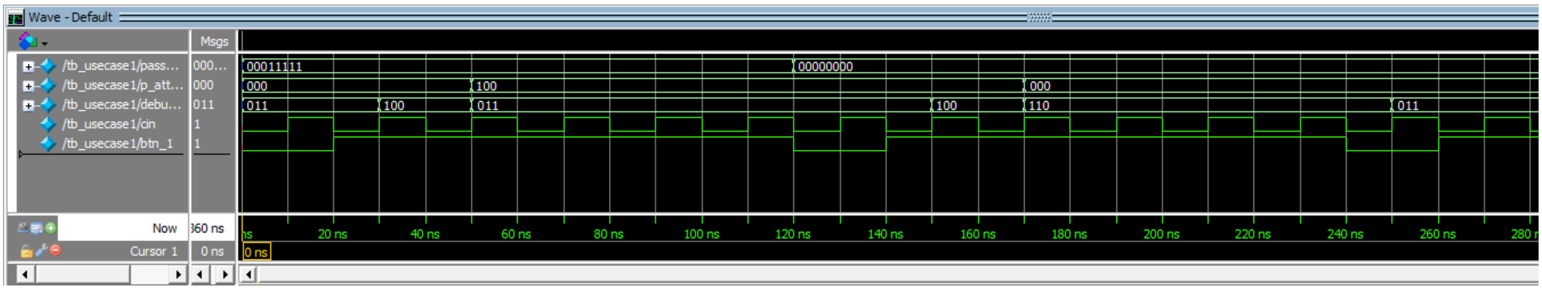
Verification

I created several end use cases for the digital safe, each representing a real world interaction scenario to cover all features and state changes. For each case, I wrote a dedicated test bench and used ModelSim to simulate waveforms. I verified the correct transitions between states (debug lights output shows state).

** In the top level module, I changed the clock divider to a faster pulse to make the simulation faster. Otherwise, the 100ms main clock cycles were taking too long to simulate.

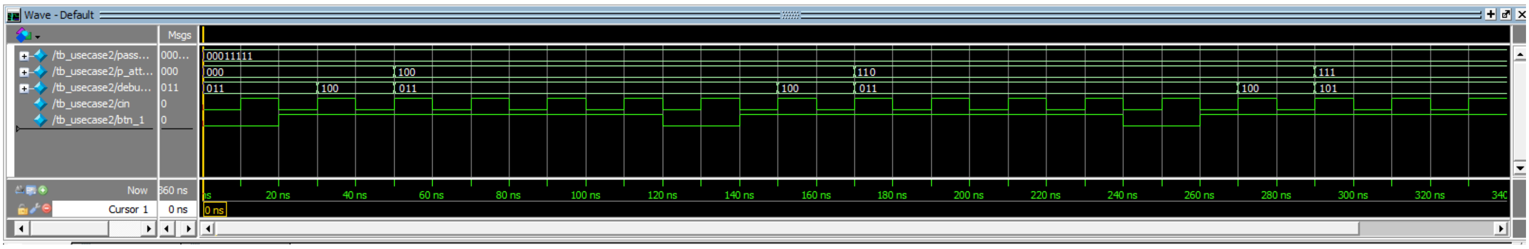
Use case 1: User enters password to unlock safe, then closes it

- User enters incorrect password and submits using btn1
  - ○ Password is invalidated (from PASSWORD_VALIDATION mode, 100)
  - ○ Password attempts counter increments by 1
- User enters correct password using input switches and submits using btn1
  - ○ Password is validated (PASSWORD_VALIDATION mode, 100)
  - ○ Password attempts is reset to 0
  - ○ State changed to OPEN_IDLE (110)
  - ○ Password attempts counter resets to 0
- User presses close button (btn1)
  - ○ State changed to IDLE (011)



Use case 2: User enters incorrect password 3 times

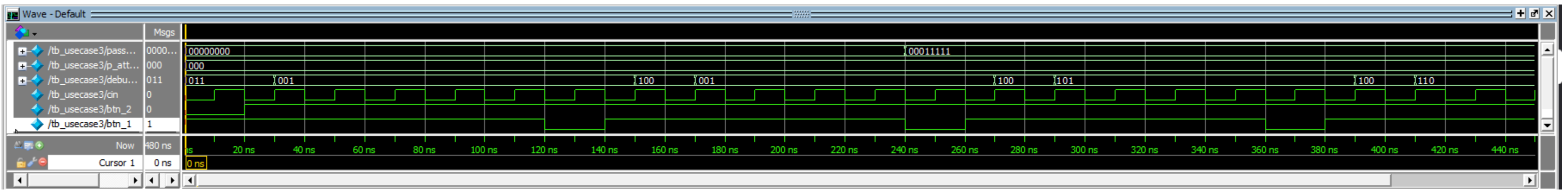
- User enters incorrect password and submits using btn1 x3
  - ○ Password is invalidated by PASSWORD_VALIDATION mode (100)
  - ○ Password attempts display increments by 1 and state returns to IDLE (011)
  - ○ User is locked out after 3 failed attempts (LOCKOUT MODE 101)



Use case 3: User changes password, and opens using new password

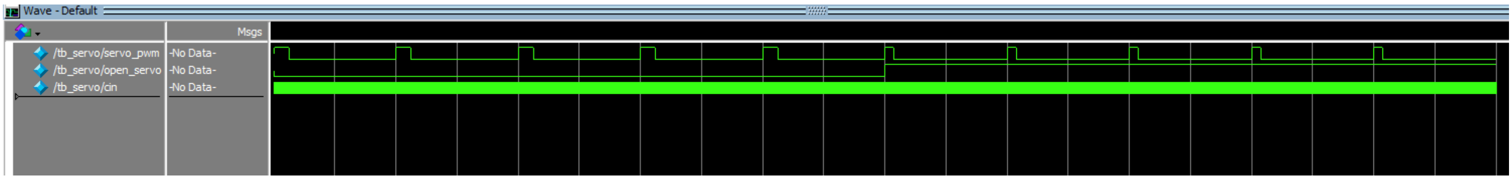
- User presses btn 2 to initiate password change
  - ○ Mode is switched to PASWORDCHANGE_OLDPASS (001)
- User enters the correct password, and presses btn 1 to verify

    - Password is validated (PASSWORD_VALIDATION mode, 100)
    - Mode is swapped to allow user to enter new password (PASSWORDCHANGE_NEWPASS, 010)
- User enters new password and presses btn 1 to save
    - New password is saved in register
- User uses switch input to enter new password and presses btn1 to submit
    - Password is validated (100, validation mode)
    - Switches to OPEN_IDLE mode, 110



<u>Module testing:</u>

- The use cases above covered most of the features, including all state changes.
- Hexadecimal displays were tested by sight. I ensured that:
    - “Open” was displayed when the servo was open
    - A countdown was displayed in lockout mode
    - User input was displayed in hexadecimal, otherwise
- The Servo controller module was tested separately, to make sure 1.5 ms clock pulses were generated when open_servo was active, and 2.5 ms pulses when open_servo was 0.



Result

The digital safe system functioned reliably across all defined use cases. The servo consistently responded to correct password entries and transitioned between locked and unlocked states. The 7 segment display accurately displayed user inputs and the countdown timer during lockout. State transitions occurred as expected based on button inputs and internal logic.

<u>Performance</u>

Servo movement responded in under 0.5 seconds to input. Display latency was not noticeable. PWM signals were accurate, stable, and moved servo to correct positions. Lockout mode successfully blocked out all inputs. Reset successfully and reliably reinitialized system.

<u>Possible Improvements</u>

More user feedback (i.e. use displays to show when to enter an old password, or if a new password was saved or cancelled) would make the product more user friendly. By using a selector MUX, with the current state as a 3 bit selector, the hexadecimal output can be changed based on the current state. This addition could further improve and optimize the current hexadecimal display implementation, and make for a cleaner solution.

Challenges

- Button bouncing: Fast periods caused buttons to register presses across multiple clock cycles, leading to unpredictable behaviours. A human-like input delay of 100ms was created for the main_clock through the clock divider to solve that problem. Another problem arose that button states were not maintained long enough to reach all logic within the loop- the solution was to use registers to maintain the state of the button at the beginning of the clock pulse throughout the whole cycle, and only take one button input at a time.
- Clock timing: Initial issues with timing the countdown accurately were fixed by deriving a slower clock from the main clock for the lockout timer. Rather than deriving a second clock divider module, the main clock was instead further divided into 1 second increments within the main loop.
- Servo PWM generation: Stability issues and unpredictable behaviours rose from using the same clock to generate PWM signals for the servo, as well as the main state machine. Since the servo would be functioning on its own loop, the second 50MHz clock on the board was used to solve this problem.

Resource

The digital safe design uses less than 1% of the DE10-Lite board's available logic and memory resources. Only 412 / 49,760 (< 1 %) logic elements are used, with the majority being used for combinational logic. Register usage is small, at 112 out of 51,509, and no block RAMs, multipliers, or PLLs are used.

The design uses 32 out of 3,110 logic array blocks (1%) and uses 52 I/O pins (14%). It employs 3 global clocks (15% of available), with low interconnect and fan out usage, which means that there is less routing complexity.

In terms of inputs and outputs, 2/3 of the clocks, both of the buttons, and almost all of the switches and LEDs on the board are being used, as well as most of the 7 segment displays. These resources may limit any future enhancements, such as increasing password length, messages on the display, adding button controls, or creating other light codes.

All in all, the implementation of the digital safe is fairly lightweight and leaves room for future enhancements (such as more user feedback features).

Resource	**Usage**
Total logic elements	412 / 49,760 (< 1 %)
-- Combinational with no register	300
-- Register only	14
-- Combinational with a register	98
Logic element usage by number of LUT inputs	
-- 4 input functions	133

-- 3 input functions	82
-- <=2 input functions	183
-- Register only	14
Logic elements by mode	
-- normal mode	277
-- arithmetic mode	121
Total registers*	112 / 51,509 (< 1 %)
-- Dedicated logic registers	112 / 49,760 (< 1 %)
-- I/O registers	0 / 1,749 (0 %)
Total LABs: partially or completely used	32 / 3,110 (1 %)
Virtual pins	0
I/O pins	52 / 360 (14 %)
-- Clock pins	3 / 8 (38 %)
-- Dedicated input pins	1 / 1 (100 %)
M9Ks	0 / 182 (0 %)
UFM blocks	0 / 1 (0 %)
ADC blocks	0 / 2 (0 %)
Total block memory bits	0 / 1,677,312 (0 %)
Total block memory implementation bits	0 / 1,677,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 288 (0 %)
PLLs	0 / 4 (0 %)
Global signals	3
-- Global clocks	3 / 20 (15 %)
JTAGs	0 / 1 (0 %)

CRC blocks	0 / 1 (0 %)
Remote update blocks	0 / 1 (0 %)
Oscillator blocks	0 / 1 (0 %)
Impedance control blocks	0 / 1 (0 %)
Average interconnect usage (total/H/V)	0.2% / 0.2% / 0.2%
Peak interconnect usage (total/H/V)	3.0% / 3.0% / 3.0%
Maximum fan-out	56
Highest non-global fan-out	49
Total fan-out	1602
Average fan-out	2.49