ĐẠI HỌC QUỐC GIA HÀ NỘI – TRƯỜNG ĐẠI HỌC CÔNG NGHỆ  
---------------------------------------------  
  
Logo, icon, company name

Description automatically generated  
  
BÁO CÁO MÔN HỌC  
Môn: Phát triển hệ thống nhúng bảo mật  
Đề tài:   
Xây dựng bộ tạo số ngẫu nhiên

|  |  |  |
| --- | --- | --- |
| Giảng viên hướng dẫn: | Thầy Bùi Duy Hiếu |  |
| Sinh viên thực hiện: | Lê Toàn Thắng | 21021441 |
|  | Phạm Thái Dương | 21021407 |
|  | Phạm Minh Quang | 21020601 |
|  |  |  |
|  |  |  |
| *Hà Nội, ngày 27 tháng 12 năm 2024* | | |

# A. ĐẶT VẤN ĐỀ, GIỚI THIỆU PHƯƠNG PHÁP

## I. Các Mối Đe Dọa An Ninh Mạng

Các mối đe dọa an ninh mạng là những hành động của các cá nhân có ý đồ xấu, nhằm đánh cắp dữ liệu, gây thiệt hại hoặc làm gián đoạn các hệ thống máy tính. Các mối đe dọa này có thể đến từ nhiều nguồn khác nhau, bao gồm các quốc gia thù địch, nhóm khủng bố, hacker cá nhân, hoặc thậm chí những người trong tổ chức (như nhân viên hoặc nhà thầu) lạm dụng quyền truy cập của họ.

## II. Các Nguồn Cơ Bản Của Các Mối Đe Dọa An Ninh Mạng

1. Các quốc gia thù địch: Tấn công vào các công ty hoặc tổ chức trong nước nhằm gây rối loạn và gây thiệt hại.
2. Tổ chức khủng bố: Tấn công cơ sở hạ tầng quan trọng, đe dọa an ninh quốc gia và gây thiệt hại cho nền kinh tế.
3. Nhóm tội phạm: Các nhóm hacker tổ chức tấn công nhằm mục đích kiếm lợi từ việc ăn cắp thông tin hoặc thực hiện các cuộc lừa đảo trực tuyến.
4. Hacker cá nhân: Tấn công vì động cơ cá nhân, sự trả thù, hoặc vì lợi ích tài chính.
5. Nhân viên hoặc đối tác có ý đồ xấu: Những người trong tổ chức sử dụng quyền truy cập của mình để đánh cắp dữ liệu hoặc gây thiệt hại.

## III. Các Loại Mối Đe Dọa An Ninh Mạng

1. Tấn công Malware (Phần mềm độc hại): Bao gồm các virus, sâu (worms), Trojan, ransomware, spyware... Các phần mềm này có thể thu thập dữ liệu nhạy cảm, tấn công vào hệ thống hoặc làm tê liệt hệ thống.
2. Tấn công Social Engineering (Kỹ thuật tấn công qua con người): Lừa đảo người dùng để cung cấp thông tin nhạy cảm hoặc cài đặt phần mềm độc hại.
3. Tấn công Man-in-the-Middle (MitM): Can thiệp vào các cuộc giao tiếp giữa hai bên để nghe lén hoặc giả mạo một trong các bên.
4. Tấn công Denial-of-Service (DoS): Làm quá tải hệ thống bằng lưu lượng truy cập cao, khiến hệ thống không thể hoạt động bình thường.
5. Tấn công Injection (Chèn mã độc vào hệ thống): Lợi dụng các lỗ hổng bảo mật trong mã nguồn của ứng dụng web để chèn mã độc vào hệ thống.

## IV. Phương pháp đề xuất

### Sử dụng bộ tạo số ngẫu nhiên (True Random Number Generator)

Với sự phát triển mạnh mẽ của các thiết bị kết nối và các cuộc tấn công ngày càng tinh vi, nhu cầu bảo mật trong các sản phẩm và hệ sinh thái trở nên quan trọng hơn bao giờ hết. True Random Number Generators (TRNGs), hay các bộ sinh số ngẫu nhiên thực sự, là yếu tố cốt lõi trong các hệ thống bảo mật, đóng vai trò quan trọng trong việc tạo ra các khóa mã hóa và bảo vệ dữ liệu nhạy cảm. Những số ngẫu nhiên yếu hoặc có thể dự đoán sẽ tạo điều kiện cho các cuộc tấn công, từ đó làm lộ khóa, đánh cắp dữ liệu và xâm nhập vào thiết bị.

### Định nghĩa về TRNG

TRNG là một thiết bị hoặc chức năng sử dụng một hiện tượng vật lý không thể đoán trước (nguồn entropy) để tạo ra dữ liệu không xác định, chẳng hạn như một chuỗi số ngẫu nhiên, phục vụ cho các thuật toán bảo mật. Các TRNG đóng vai trò quan trọng trong việc tạo và bảo vệ các bí mật (như khóa mã hóa) trong các thiết bị kết nối, và tạo thành một "chuỗi tin cậy" từ SoC (System on Chip) đến các lớp ứng dụng và truyền thông đám mây. Sự bảo mật của hệ thống phụ thuộc vào độ ngẫu nhiên của số được tạo ra bởi TRNG.

### Các tiêu chuẩn dành cho TRNG

Có một số tiêu chuẩn quốc tế xác định các phương pháp kiểm tra và chứng nhận độ ngẫu nhiên của TRNGs, bao gồm:

* NIST SP 800-90A/B/c: Các tiêu chuẩn này xác định các tiêu chí phân tích thống kê mà một RNG cần phải đáp ứng để được coi là đủ ngẫu nhiên cho các ứng dụng mã hóa.
* AIS 20/31 của BSI: Tiêu chuẩn của Đức cho RNGs.
* Các chứng nhận như FIPS 140-2/140-3, Common Criteria (CC), và OSCCA của Trung Quốc đảm bảo rằng sản phẩm cuối cùng đáp ứng các yêu cầu bảo mật và có thể được chứng nhận cho việc sử dụng trong các ứng dụng chính thức.

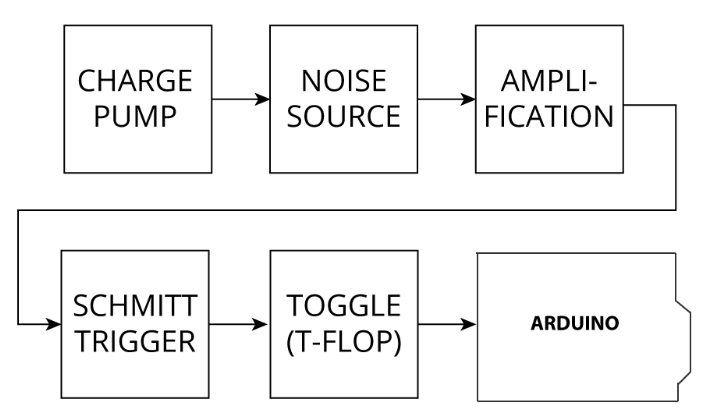
### Ứng dụng của TRNG

True Random Number Generators (TRNGs) có nhiều ứng dụng quan trọng trong các hệ thống bảo mật và các lĩnh vực công nghệ khác. Dưới đây là một số ứng dụng phổ biến của TRNG:

* Mã hóa
* Tạo khóa bảo mật
* Xác thực và chứng thực
* Giai mã và xác minh

# B. THIẾT KẾ MẠCH TẠO SỐ NGẪU NHIÊN DỰA TRÊN NHIỄU AVALANCHE

I. Sơ đồ khối của mạch điện



*Hình 1. Sơ đồ khối mạch tạo nhiễu Avalanche*

II. Nguyên lý hoạt động của các khối

a. Charge pump: khối này sẽ thực hiện nhiệm vụ tăng điện áp đầu vào từ 5V lên 12V để cung cấp điện áp cho các khối khác trong mạch.

b. Noise source: đây là khối sẽ thực hiện việc tạo nhiễu avalanche để phục vụ quá trình sinh số ngẫu nhiên.

c. Amplification: khi tín hiệu nhiễu ngẫu nhiên ban đầu đi qua tầng này sẽ được khuếch đại đến một ngưỡng nhất định giúp tín hiệu dễ dàng được xử lý.

d. Schmitt Trigger: tín hiệu nhiễu ngẫu nhiên ban đầu sau khi được khuếch đại sẽ được chuyển sang dạng xung ô vuông kỹ thuật số.

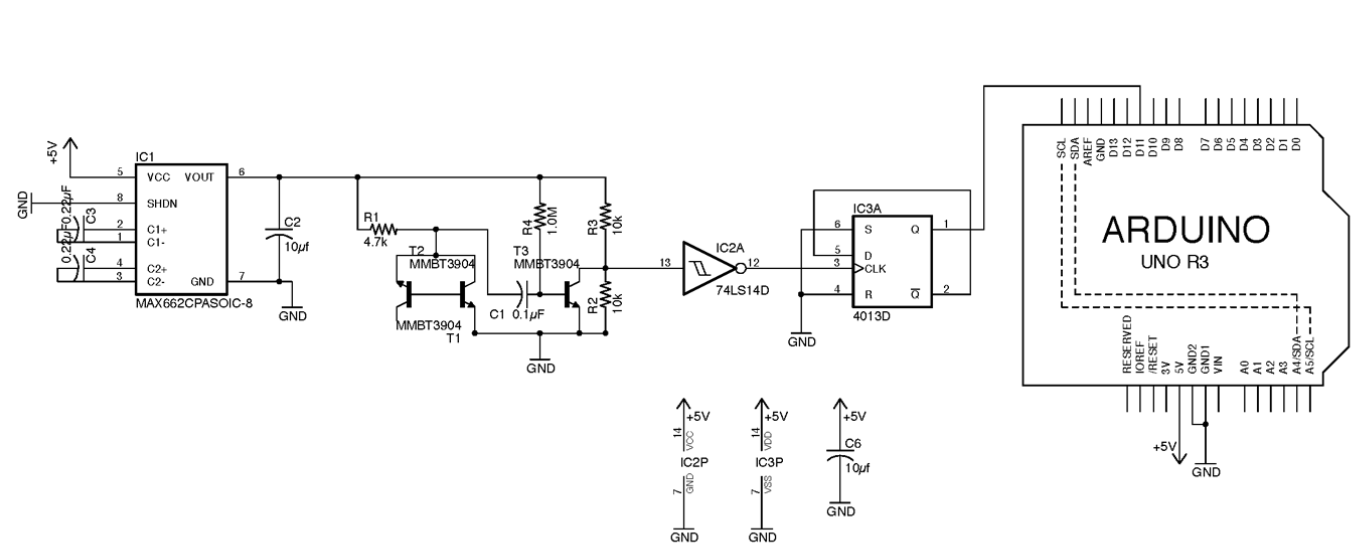
e. Toggle (T-Flop): khối này hoạt động như một công tắc chuyển đổi giữa giá trị cao (1) và thấp (0) để chuyển đổi xung sang dạng tín hiệu 1-0 và đưa vào Arduino xử lý.

III. Các linh kiện cần chuẩn bị

|  |  |  |
| --- | --- | --- |
| Linh kiện | Miêu tả | Số lượng |
| SN74LS14N | Hex-Schmitt Trigger | 1 |
| CD4013BE | D-Flop | 1 |
| 22uF Cap |  | 2 |
| 0.1uF Cap |  | 1 |
| 1uF Cap |  | 2 |
| 10uF Cap |  | 2 |
|  |  |  |
| 2N3904 | NPN Transistor | 3 |
| 4.7Kohm Res |  | 1 |
| 1M Res |  | 1 |
| 10K Res |  | 2 |
| Arduino |  | 1 |

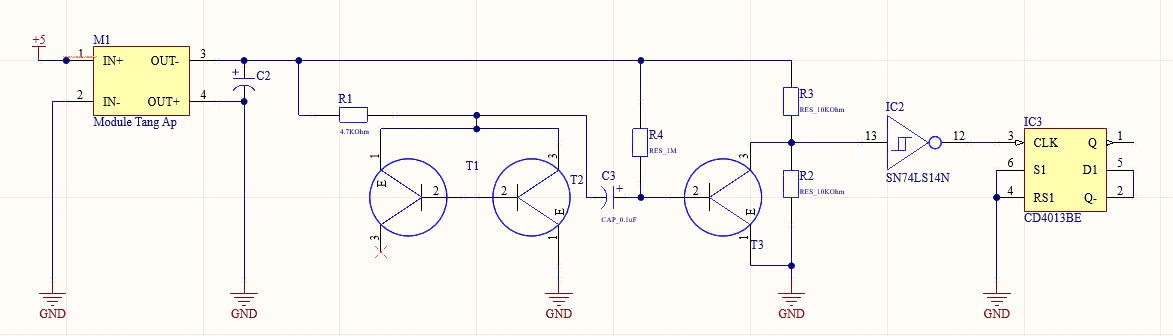
IV. Thiết kế mạch nguyên lý trên phần mềm Altium

1. Mạch đề xuất



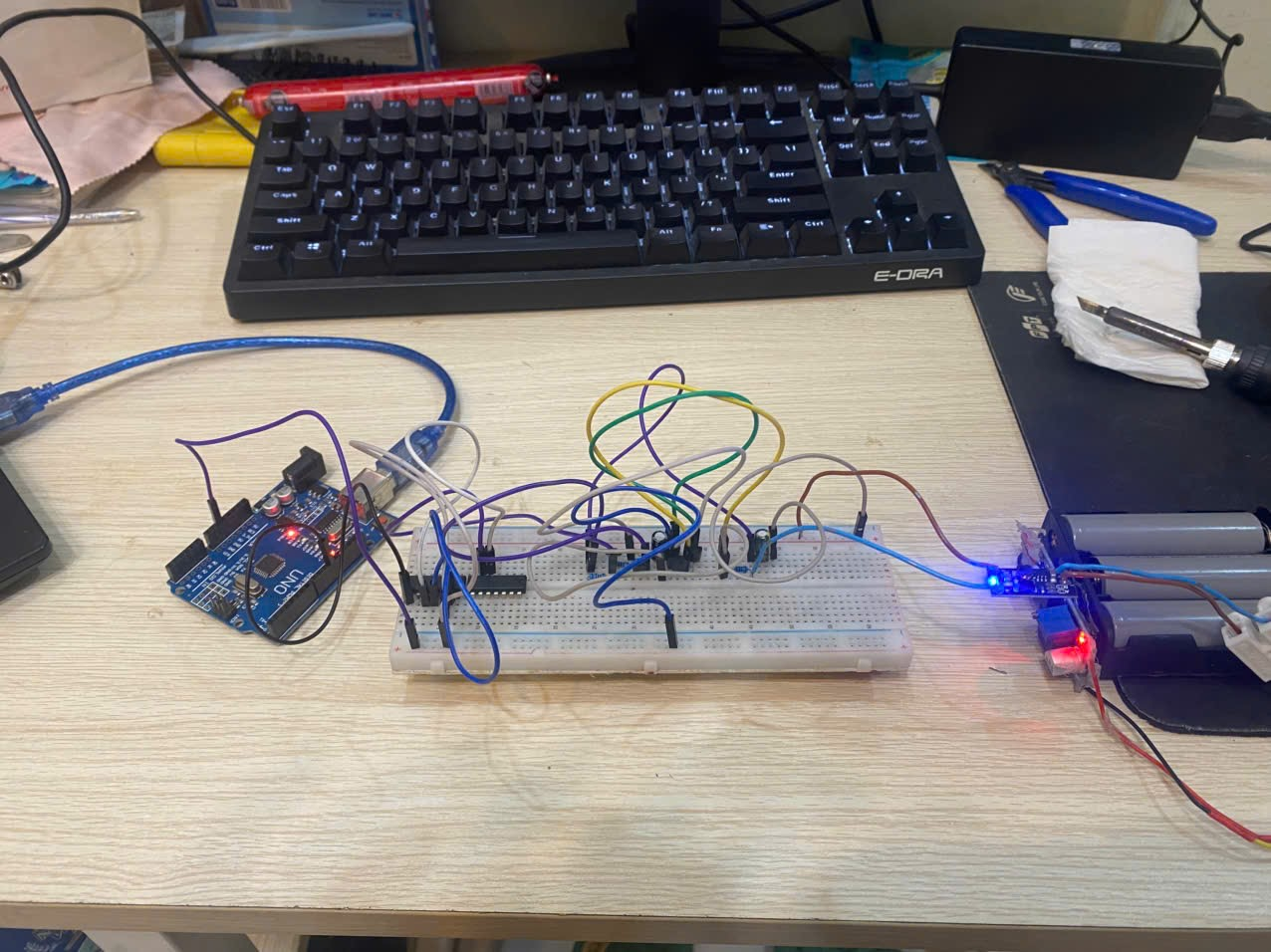
*Hình 2. Mạch nguyên lý được đề xuất*

2. Mạch thiết kế lại



*Hình 3. Mạch nguyên lý được thiết kế trên Altium*

3. Sản phẩm sau khi thiết kế



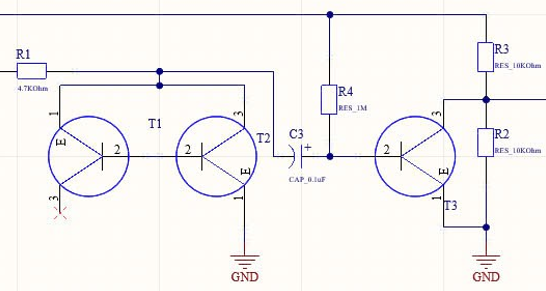
*Hình 4. Mạch sản phẩm*

V. Cơ chế hoạt động của các khối trong mạch

1. Khối tăng áp

Module tăng áp-boost converter có chức năng tăng điện áp một chiều ở đầu vào thành điện áp một chiều có giá trị cao hơn ở đầu ra. Tụ điện được đặt ở đầu ra của module có chức năng ổn định điện áp đầu ra ở mức 12V.

2. Khối tạo nhiễu Avalanche



*Hình 5. Khối tạo nhiễu Avalanche trong mạch*

a. Giới thiệu về cách hoạt động của mạch

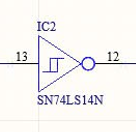
Mạch hoạt động dựa trên nguyên lý nhiễu avalanche tạo bởi transistor T1 và T2. Tín hiệu nhiễu sau khi được tạo ra sẽ được khuếch đại bởi transistor T3.

b. Nguyên lý nhiễu avalanche

Khi đặt điện áp ngược vượt quá điện áp đánh thủng lên diode hoặc lớp bán dẫn p-n, dòng điện tăng đột ngột do các điện tử va chạm với nguyên tử, tạo ra thêm cặp điện tử-lỗ trống. Hiện tượng này tạo ra nhiễu avalanche – một loại nhiễu có tính ngẫu nhiên cao.

Trong thực tế, người ta thường sử dụng phân lớp base-emitter của transistor NPN để tạo nhiễu avalanche, vì lớp này có điện áp đánh thủng thấp. Nhiễu này được ứng dụng làm nguồn entropy cho các hệ thống tạo số ngẫu nhiên.

3. Khối tạo xung clock



*Hình 6. Khối tạo xung clock*

Khối tạo xung clock sử dụng IC 74HC14 nhận điện áp vào là nhiễu sau khi đã khuếch đại. Nếu điện áp đầu vào của khối lớn hơn giá trị VT+ thì điện áp đầu ra sẽ được đặt ở mức thấp VOL. ngược lại, nếu điện áp đầu vào của khối nhỏ hơn giá trị VT- thì điện áp ở đầu ra sẽ được đặt ở mức cao VOH.

4. Khối D-Flip Flop

Khối flip-flop hoạt động với chân Q xuất tín hiệu ra và chân Q̅ là tín hiệu đảo của Q, được nối vào chân dữ liệu vào D. Khi có xung clock đưa vào chân CLK, trạng thái của các chân thay đổi theo sườn lên. Cụ thể, khi Q được đặt ở mức 1 thì Q̅ sẽ ở mức 0. Khi xung clock có sườn lên, Q chuyển về mức 0 và Q̅ thành mức 1. Ở sườn lên tiếp theo, hai chân lại đảo trạng thái, Q trở về mức 1 và Q̅ về mức 0. Quá trình này diễn ra liên tục, và Arduino sẽ đọc giá trị đầu ra tại chân Q để sử dụng trong hệ thống.

*Hình 7. Khối D-Flip Flop*

VI. Bảng linh kiện được sử dụng

|  |  |  |
| --- | --- | --- |
| Tên linh kiện | Mô tả | Số lượng |
| Module tăng áp 5V-12V |  | 1 |
| SN74LS14N | Hex-Schmitt Trigger | 1 |
| CD4013BE | D-Flop | 1 |
| 0.1uF Cap | Tụ 0.1 microFarad | 1 |
| 2N3904 | Transistor loại NPN | 3 |
| 10uF Cap | Tụ 10 microFarad | 1 |
| 1M resistors | Điện trở 1M Ohm | 1 |
| 4.7k resistors | Điện trở 4.7K Ohm 1/4W 1% | 1 |
| 10k resistors | Điện trở 10k Ohm | 2 |
| Breadboard | Mạch test | 1 |
| Dây nối |  |  |
| Arduino UNO R3 |  | 1 |

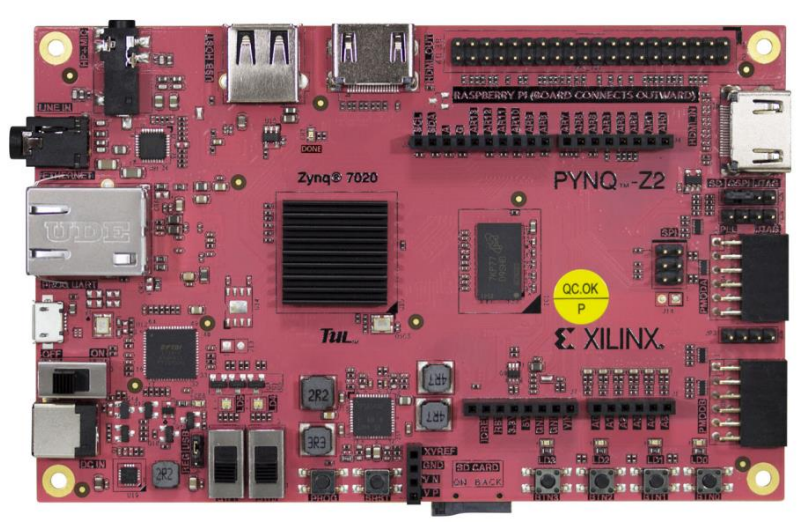
*Bảng 1. Các linh kiện*

# C. THỰC HIỆN TRÊN FPGA

I. Giới thiệu

FPGA (Field Programmable Gate Array) là một loại vi mạch tích hợp có thể lập trình để thực hiện các chức năng logic phức tạp. Không giống như các vi điều khiển truyền thống, FPGA cho phép thiết kế phần cứng tùy chỉnh bằng cách cấu hình các khối logic và kết nối nội bộ. FPGA nổi bật với khả năng xử lý song song, hiệu suất cao và độ linh hoạt, phù hợp cho các ứng dụng trong xử lý tín hiệu, viễn thông, và trí tuệ nhân tạo.

Kit PYNQ-Z2 là một nền tảng phát triển dựa trên FPGA Xilinx Zynq-7000, tích hợp cả mạch logic lập trình và bộ xử lý ARM Cortex-A9. Kit hỗ trợ các ứng dụng học sâu, xử lý hình ảnh, và hệ thống nhúng, đồng thời cho phép lập trình dễ dàng bằng ngôn ngữ Python thông qua môi trường PYNQ. Với thiết kế nhỏ gọn và tính năng mạnh mẽ, PYNQ-Z2 được sử dụng rộng rãi trong nghiên cứu, giáo dục và phát triển công nghệ.



*Hình 8. KIT PYNQ-Z2 EVALUATION*

II. Cấu hình phần cứng để kết nối mạch tạo số ngẫu nhiên cho PYNQ-Z2

Để thực thi bộ sinh số ngẫu nhiên, trước tiên phải thiết kế và cấu hình cho FPGA. Sau đây là các bước trong quy trình cấu hình cho FPGA để nhận dữ liệu từ mạch, và hiển thị trên màn hình máy tính. Việc thiết kế được thực hiện trên phần mềm Vivado-một công cụ phần mềm của Xilinx được thực hiện theo các bước dưới đây:

a. Tạo Block Design

- Tạo Block Design bằng cách thêm các IP và kết nối chúng.

- Bước này giúp thêm các mô-đun cần thiết để FPGA có thể nhận và truyền dữ liệu.

- Sử dụng công cụ "Validate Design" để kiểm tra thiết kế đáp ứng yêu cầu hệ thống.

b. Tạo tệp Wrapper cho hệ thống

- Tệp Wrapper được dùng để chuyển đổi Block Design sang ngôn ngữ mô tả phần cứng (HDL).

- Chọn "Create HDL Wrapper" để tạo mô hình VHDL cấp cao nhất.

c. Tạo tệp Constraint

- Tệp Constraint ràng buộc các chân đầu ra cho FPGA.

- Để tạo được Constraint cần chạy chế độ Systhesis trước. Đây là quá trình chuyển đổi ngôn ngữ mô tả phần cứng của thiết kế thành netlist (Netlist là một danh sách các cổng và các cấu trúc logic cụ thể trong thiết kế). Bước này còn giúp điều chỉnh và tối ưu hóa thiết kế trước khi thực thi trên FPGA.

- Sau đó ánh xạ các chân cần sử dụng và xuất ra file XDC (constraint).

d. Implementation: Bước này giúp triển khai thiết kế từ mô hình logic thành nguyên mẫu để chạy trên FPGA. Quá trình này để đảm bảo thiết kế được thực thi một cách chính xác trên FPGA.

e. Tạo Bitstream: Đây là một trong những bước quan trọng nhất trong quá trình cấu hình FPGA. Tạo một tệp bitstream để có thể nạp vào FPGA để thực hiện chức năng cụ thể đã được thiết kế.

f. Lập trình trên FPGA: Sau khi tạo tệp bitstream từ bước trước đó, cần tạo code trên phần mềm SDK. Chương trình sẽ được biên dịch và chạy trên FPGA, kết quả sẽ được hiển thị trên phần mềm.

*Hình 9. Cấu hình phần cứng của PYNQ-Z2*

Sau khi đã cấu hình phần cứng trên PYNQ-Z2, tiếp theo là quá trình kết nối và giao tiếp giữa mạch sinh số ngẫu nhiên và bo mạch PYNQ-Z2.

Dữ liệu từ bộ sinh số ngẫu nhiên sẽ được truyền đến bo mạch PYNQ-Z2 qua chân IO A5. Để dữ liệu có thể truyền, cần thêm các ràng buộc GPIO tại tệp XDC trong quá trình cấu hình. Cần đảm bảo rằng kết nối này được thực hiện đúng cách để dữ liệu có thể truyền đến mạch PYNQ-Z2.

PYNQ-Z2 giao tiếp với máy tính thông qua cổng UART. Vì vậy, cần kết nối các chân UART của PYNQ-Z2 với máy tính và cấu hình AXI-UART trong thiết kế. Cổng UART được sử dụng để gửi và nhận dữ liệu giữa FPGA và máy tính.

Khi PYNQ-Z2 được kết nối với máy tính, dữ liệu đầu ra từ mạch sinh số ngẫu nhiên có thể được truyền và hiển thị trên màn hình máy tính. Dữ liệu được biểu diễn dưới dạng các chuỗi bit 0 và 1, sử dụng phần mềm PUTTY để dữ liệu hiển thị rõ ràng hơn.

*Hình 10. Mạch sinh số ngẫu nhiên kết nối với PYNQ-Z2*

III. Dữ liệu đầu ra

IV. Kết luận