Universitatea Tehnică din Cluj-Napoca  
Facultatea de Automatică și Calculatoare  
Catedra de calculatoare

*Comunicarea dintre placa Basys 3 si un Mouse printr-un port USB*

**Realizator: Stroia Lucian Dorin**

**Grupa: 30237**

**Disciplină: Structura Sistemelor de Calcul**

**Îndrumător de proiect: Beliga Octavian**

**Data:13.01.2020**

**Cuprins**

1. Rezumat…………………………………….................3
2. Introducere…………………………………………….3
3. Fundamentare teoretică………………………………..5
4. Proiectarea si implementarea………………………….8
5. Rezultatele experimentale……………………………12
6. Concluzii……………………………………………..14
7. Bibliografie…………………………………………..15

1.Rezumat

Acest proiect are ca și scop reprezentarea mișcărilor unui mouse, aceste mișcări fiind descrise prin intermediul unui set de biți care trebuie interpretați. Legătura dintre componentele descrise pentru detectarea mișcărilor mouse-ului este reprezentată de un FPGA, care dispune de un port USB, la care este conectat mouse-ul.

Odată ce datele trimise de către mouse sunt recepționate, se realizează trimiterea acestora către portul COM, de unde, prin intermediul altei aplicații, vor fi interpretate și va comunica vizual cu utilizatorul, indicând schimbările de stare ale mouse-ului prin intermediul unui Frame, unde se va reprezenta o săgeată de mouse care își schimbă poziția în funcție de mișcările acționate asupra mouse-ului. De asemenea, va fi posibilă si detecția acțiunii asupra cele 3 butoane standard ale mouse-ului.

2.Introducere

* 1. Descrierea proiectului

Proiectul ales pune in evidență funcționalitatea plăcii Basys 3, care trebuie sa asigure comunicarea cu un mouse printr-un port USB. Această comunicare trebuie să se realizeze prin transmiterea datelor, recepționate de la mouse, prin intermediul unei componente descrise în limbajul VHDL.

Punerea în evidență a acestui tip de comunicare va fi realizată prin intermediul unei aplicații care o să interpreteze datele recepționate de la mouse, aceste date fiind reprezentate de seturi de biți, care au fost formate în cadrul programului VIVADO.

Astfel, prin intermediul unui frame, utilizatorul va putea observa mișcările realizate prin intermediul mouse-ului, fiind realizata si o aplicație grafică care să se ocupe de acest aspect

* 1. Domeniul de studiu

Domeniul de studiu are la bază interfața de mouse PS/2, care folosește același protocol

ca și tastatura PS/2. Acest standard a apărut inițial în manualul de referință tehnic IBM.

Mouse-ul standard PS/2 acceptă următoarele intrări:

1. Mișcarea X(dreapta/stânga)
2. Mișcarea Y(dreapta/stânga)
3. Butonul stâng
4. Butonul drept
5. Butonul din mijloc

Mouse-ul citește aceste intrări la o frecvență regulată și actualizează diverse contoare și steaguri pentru a reflecta stările de mișcare și butoane. Există multe dispozitive de indicare PS / 2 care au intrări suplimentare și pot raporta date diferit.

* 1. Principalele obiective

Datele primite de la mouse prin intermediul plăcuței Basys3 trebuie decodificate

astfel încât să se observe în model și să se facă o diferență între semnalele indicate de către mouse la diferite momente de timp. Astfel, au fost realizate pachete de câte 8 biți care să pună în evidență starea mouse-ului, mai exact poziția și daca a fost realizată o acțiune asupra celor 3 butoane standard ale unui mouse.

* 1. Soluția aleasă

Au fost utilizate mai multe componente care să contureze datele care trebuie trimise către aplicația python. Astfel, în soluția aleasă de mine sunt integrate componente descrise in VHDL precum FSM, pentru a detecta schimbarea stării mouse-ului la un anumit moment și de a trasa drumul datelor pentru a fi prelucrate, dar și o componentă responsabilă de transmiterea datelor către portul COM, numită UART.

* 1. Secțiunile următoare

În continuare, vor fi prezentate detalii legate de tehnologiile folosite și de componentele periferice utilizate pentru realizarea proiectului. Vor fi explicate în amănunt fiecare componentă descrisă în programul Vivado, modul în care au fost proiectate și implementate acestea pentru o bună funcționare a întregului proiect.

La final, sunt puse în evidență rezultatele obținute în urma realizării unei simulări dar și din timpul funcționării propriu-zise ale proiectului.

3.Fundamentare teoretică

*3.1* Vivado

Pentru realizarea proiectului, principalul program este reprezentat de programul Vivado, care suporta caracteristicile limbajului de programare hardware iar pentru realizarea aplicației care va pune în evidență mișcările mouse-ului va fi folosit un limbaj de programare software Python.

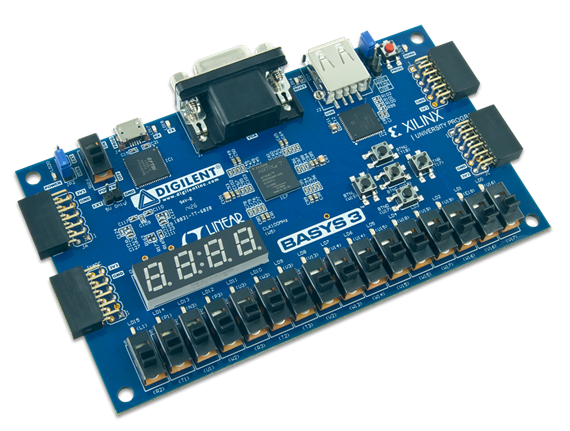
Vivado este un program creat de către compania Xilinx, fiind realizat in scopul sintezei si analizei proiectelor, vine cu îmbunătățiri față de programul Xilinx ISE deoarece are o serie de funcții suplimentare destinate dezvoltării sistemelor de cipuri și a sintezei de nivel înalt

Acest program a fost descris de către utilizatori ca fiind “bine conceput, bine integrat, aprins rapid, scalabil, întreținător și intuitiv“.

Legătura care asigură comunicarea dintre mouse și programul Vivado a fost realizată prin intermediul utilizării unei plăcuțe Basys3, un FPGA destinat exclusiv pentru programul Vivado.

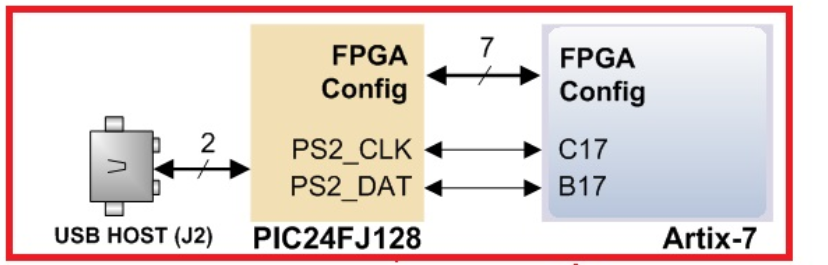
3.2 Placa Basys3

Placa Basys 3, este o platformă de pregătire a circuitului digital bazată pe cea mai recentă Artix-7 FPGA. Are un cost relativ redus, dispune de porturi USB, VGA, si poate susține proiecte care variază de la circuite combinaționale până la circuite secvențiale complexe.Această plăcuță include 16 switch-uri, 16 leduri, si alte dispozitive de intrare si ieșire pentru realizarea unui număr mare de proiecte fără a fi necesară introducerea altor dispozitive suplimentare.



3.3 Protocolul PS2

Așa cum este ilustrat în imaginea de mai jos, semnalele PS2\_CLK și PS2\_DAT ale microcontrolerului sunt utilizate pentru a implementa interfață PS / 2 pentru comunicarea cu mouse-ul.

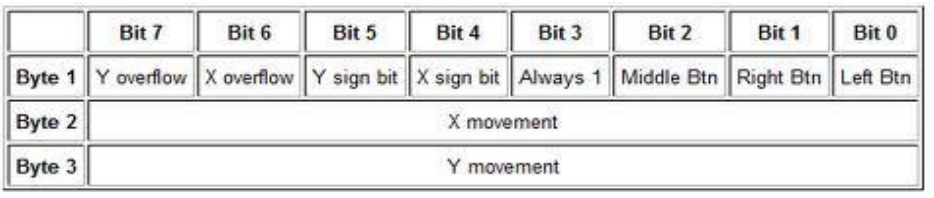


În acest caz, microcontrolerul se comportă ca un bus PS / 2, iar un mouse care este conectat la portul USB-HID poate folosi bus-ul serial cu două fire (ceas și date) (protocol PS / 2) pentru a comunica cu FPGA.Mouse-ul trimite semnale de ceas și date către FPGA atunci când este mutat.

Descrierea in program a stării mouse-ului va fi realizată prin intermediul unui pachet de 33 de biți.

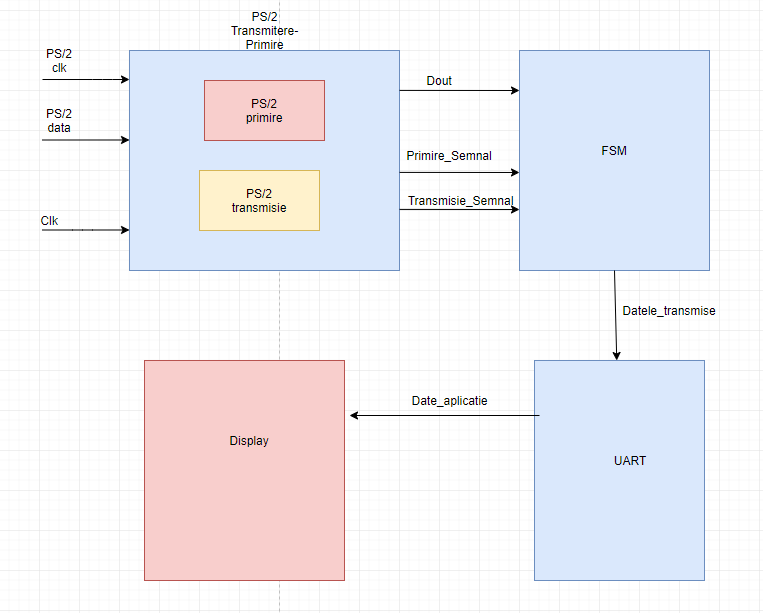
Datele de mișcare includ un octet de stare al mouse-ului, octetul direcției X și octetul direcției Y. Datele sunt valabile numai la marginea care coboară a semnalului de ceas. Octeții direcție X și direcție Y sunt utilizați pentru a afișa coordonata relativă a mouse-ului, care deplasându-se la dreapta / stânga generează X pozitiv / negativ, iar deplasarea în partea de sus / jos generează Y pozitive / negative.

Biții XS și YS în octetul de stare al mouse-ului este folosit ca biți de „semn” ai X și Y. Cu cât X și Y sunt mai mari, cu atât viteza mouse-ului este mai rapidă. Biții L și R din octetul de stare al mouse-ului sunt folosiți pentru a indica dacă este apăsat butonul stâng și drept



4.Proiectare și implementare

4.1 Schema bloc



Pentru proiectarea și implemetarea proiectului, au fost descrise mai multe componente, fiecare dintre acestea urmând să fie prezentate mai în detaliu în această secțiune.

4.2 Partea de VHDL

4.2.1 Componenta Debouncer

Această componentă a fost creată deoarece atunci când există un semnal a care variază între mai multe valori, la o schimbare dintr-o anumită valoare la o altă valoare, până să treacă la valoarea propusă, apar niște fluctuații între valoarea de low si de high. Aceste fluctuații sunt îndepărtate prin intermediul debouncer-ului, care exportă valoarea corectă a semnalului dorit în intervalul de timp ales de către utilizator.

Valoarea clock-ului specific pentru plăcuța Basys3 este egală cu 100 Mhz. Pentru setarea valorii debouncer\_counter\_size, cu care are loc port maparea la componenta Debouncer a semnalelor de ps2\_clk, si ps2\_data, după ce se realizează o sincronizare cu clock-ul programului, se calculează pe baza formulei ” 2 debounce\_counter\_size / clk\_freq = 5us ” unde clk\_freq este egal cu 100Mhz. De aici rezultă faptul că debounce\_counter\_size este egal cu 9.

4.2.2 Componenta FSM

Un automat finit sau o "mașină cu un număr finit de stări" este un model de comportament compus din stări, tranziții și acțiuni. O stare stochează informații despre trecut, adică reflectă schimbările intrării de la inițializarea sistemului până în momentul de față

Această componentă este realizată în ideea de a controla tranzacțiile mouse-ului. Conține atributele generice utile pentru port maparea debouncer-ului prezentat anterior dar și o serie de porturi de intrare și ieșire a căror semnificație va fi explicată în continuare.

Sunt prezente pe lângă caracteristicile standard ale unei astfel tip de componentă, clk și reset, și semnale precum transmisie\_enable semnal care pornește transmisia datelor recepționate în transmisie\_comanda, adică 8 biți de comandă și un bit de paritate.

Este declarat un nou tip de semnal, reprezentat prin intermediul cuvântului cheie type, care indică vectorul de stări prin care trec datele primite de la mouse. Acest vector conține stările (receive, inhibit,transmission,transmission\_complete), fiecare dintre aceste stări având câte o semnificație. Starea de receive, este starea în care programul intră într-o buclă până la primirea unei comenzi. Acest lucru este indicat de semnalul transmision\_enable, care pornește transmisia datelor. În acel moment, transmisie\_busy este setat pe valoarea logică 1, indicând faptul că se află o transmisie în acțiune. Acest semnal este utilizat pentru a bloca alte recepții de transmisie ale mouse-ului. Starea de inhibit, este o stare în care se încetinește procesul de transmisie al datelor, pentru a oferi cursivitate respectiv corectitudine în modul de primire al datelor de la mouse.

În starea de transmission, are loc transmisia propriu-zisă a datelor, fiind contorizat fiecare bit care vine de la mouse. În momentul în care numărul de biți este egal cu 11, are loc finalizarea transmisie, trecând în ultima stare, adică transmission\_complete. Dacă s-a ajuns în această stare, iar semnalele ps2\_clk\_debounced și ps2\_data\_debounced sunt diferite ambele în mod simultan de valoarea logică 1, atunci înseamnă ca transmisia nu s-a terminat, starea următoare fiind tot cea în care ne aflăm. În caz contrar, adică dacă ps2\_clk\_debounced și ps2\_data\_debounced sunt ambele 1, atunci transmisia s-a încheiat si se poate trece la o nouă transmisie, starea următoare fiind reprezentată de starea de receive, în care programul este deschis primirii unei noi transmisii de la mouse.

Tot în această componentă, a fost realizat și un proces care sincronizează clk-ul plăcuței cu clk-ul programului, pentru a oferi corectitudine și simultaneitate.

Este creat și un semnal care indică dacă există o eroare în primirea datelor, acest semnal fiind destinat verificării biților de start, stop respectiv de paritate ale datelor transmise de la mouse.

4.2.3 Componenta General

În această componentă, are loc port maparea la componenta FSM, însă conține și un proces pentru controlul stărilor în care se află mouse-ul. Practic, comunicarea cu mouse-ul se realizează asemănător cu o comunicare client-server, deoarece, în procesul din componenta General, este setat transmisie\_comandă, pentru a cere date de la mouse, iar prin intermediul semnalului ps2\_code, un vector de 8 biți, se primește răspunsul de la mouse. Mouse-ul trimite diferite ps2\_code-uri în funcție de starea în care se află. Aceste ps2\_code-uri au fost verificate iar în funcție de valoarea lor, au fost creat întreg ciclul procesului, fiind conturat un drum prin stările definite anterior. Inițial, este primit un ps2\_code care trebuie comparat cu codul, ”10101010”, cod trimis de către mouse, indicând începutul stării de recepție ale datelor de la mouse.

În starea următoare, pornirea recepției se realizează după verificarea ps2\_code-ului cu valoarea ”00000000”, pentru verificarea id-ului trimis de către mouse. În caz de egalitate, se ajunge în starea de enable, unde, în cazul în care nu există deja o comandă în acțiune, se trimite prin intermediul tx\_cmd, ” 011110100” care indică ”Enable Data Reporting”.

Se trece în starea următoare, unde este verificată dacă s-a trimis un nou ps2\_code. Dacă mouse-ul răspunde cu ps2\_code egal cu 0xFA, atunci mouse-ul este pregătit să ofere informații legate de starea sa.

În ultima stare, starea de stream, se receptionează datele de la mouse, și se formează gruparea de 24 de biți care conține date legate de apăsarea butoanelor mouse-ului, x movement, y movement, y sign bit, x sign bit, etc.

După formarea celor 3 octeți, se trece iarăși la prima stare, în cazul în care există un nou cod trimis de către mouse.

Este introdus un semnal suplimentar care indică finalizarea formării celo 3 octeți de date de la mouse. Acest semnal este format în cadrul unui proces, iar atribuirea lui la 0 se realizează atunci când se resetează număr de packet\_byte la 0.

4.2.4 Componenta Principal

În această componentă, are loc legarea propriu-zisă a întregului proiect, fiind adăugate pe lângă componeta General, încă 2 componente.

A fost folosit un bistabil D, în care au fost introduse datele primite de la mouse, adica gruparea de 24 de biți pentru a oferi cursivitate programului, și pentru a nu se pierde vreo altă grupare.

A fost folosit si componenta dezvoltată în cadrul laboratorului, numită UART, care se ocupă cu transmiterea bit cu bit a celor 24 de biți către portul COM, de unde urmează să se realizeze citirea datelor necesare aplicației realizată în python.

Un receptor-transmițător asincron universal (UART) este un dispozitiv hardware computerizat pentru comunicații seriale asincrone în care formatul datelor și viteza de transmisie sunt configurabile.

Acesta componenta este necesara pentru gestionarea actiunilor care au fost realizate prin intermediul mouse-ului, fiind posibila detectia actiunilior multiple la intervale scurte de timp, precum si schimbarea pozitiei mouse-ului.

Din cauza faptului ca UART transmite doar grupări de câte 8 biți, a fost necesară o adăugare a unei logici suplimentare, pentru realizarea unui nou FSM care coordonează ce grupare de biți să utilizeze UART-ul pentru a realiza transmiterea datelor.

Astfel, a fost descris în cadrul acestei componente încă 2 procese, un proces sensibil la clk, pentru a realiza trecerile dintr-o stare în alta, și alt proces care este sensibil la semnalul declarat anterior ”stare”, care indică în funcție de starea în care se află programul, când să se activeze semnalul de start al UART-ului, și ce grupare din cei 3 octeți să fie transmiși astfel încât să se păstreze ordinea corectă a datelor.

4.3 Partea de Python

În cadrul aplicației de Python, am folosit conceptul de Threading deoarece în urma conectării la portul COM, a fost pornită o buclă care citește date în mod continuu de pe acest port, și în același timp, s-a dorit vizualizarea rezultatelor în timp real, ceea ce presupune deschiderea ferestrei în care este reprezentat mouse-ul și actualizarea acestui Frame la fiecare detecție noua de mișcare.

Astfel, se poate observa dacă a fost apăsat unul dintre butoanele stâng, drept, sau mijloc și tot odată, o mișcare a unei imagini care reprezintă cursorul mouse-ului în timp real. Pentru identificarea activităților asupra mouse-ului, a fost preluat fiecare grupare de 24 de biți de pe portul COM, iar apoi, toată atenția a fost fixată asupra biților care indica apăsarea butoanelor.

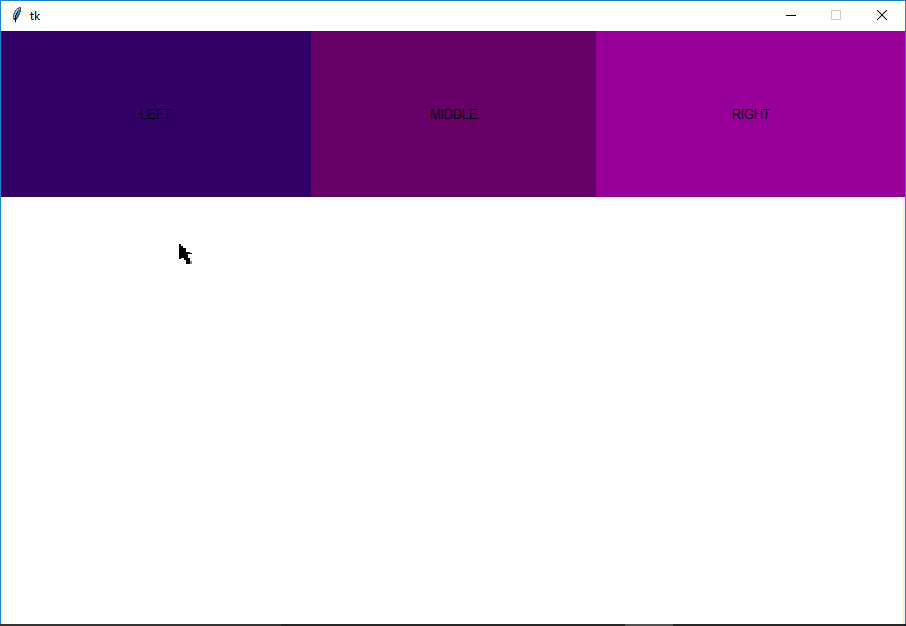
Octeții de X movement si Y movement, au fost transformați dintr-un sir de elemente binare într-un întreg, rezultând numere care indică cu cât s-a modificat x, respectiv y de la ultima schimbare a poziției mouse-ului.

1. Rezultate experimentale

În urma rulării programului celor doua programe dezvoltate, utilizatorul poate observa la fel

ca și în imaginea atașată mai jos frame-ul care se deschide odată după pornirea programelor, fiind posibilă testarea corectitudinii, prin intermediul acțiunilor asupra mouse-ului.

Odată ce apare o acțiune asupra mouse-ului, de exemplu apăsarea unui buton din cele standard care dispune, atunci unul dintre cele 3 panel-uri își va schimba culoarea in culoarea roșie. Dacă este o apăsare continuă și există și o modificare a poziției mouse-ului, atunci panelul care corespunde butonului apăsat își schimba culoarea în roșu iar imaginea cursorului se modifică în funcție de acțiunile utilizatorului.



.

Se poate observa și reprezentarea a celor 3 octeți de date trimiși de către mouse, prin intermediul ledurilor de pe plăcuța Basys3, fiind posibilă mișcarea prin cei 3 octeți prin intermediul primelor 2 switch-uri.

În imaginea de mai jos, se poate observa ca cei 8 biți din partea dreaptă sunt activi, ceea

ce indică faptul ca X este egal cu 255.





1. Concluzii

În concluzie, realizarea acestui proiect a implicat atât o parte de descriere hardware,

cât o realizare a unei aplicații în python, combinând modul de a obține și de a decodifica datele de la mouse cu o reprezentare în timp real a coordonatelor acestuia printr-o imagine care reprezintă cursorul.

Pentru o întelegere mai bună a funcționalității proiectului, utilizatorul poate să trateze mouse-ul ca fiind un dispozitiv, care oferă informații legate de poziția și acțiunea asupra butoanelor doar dacă a primit anumite coduri pentru o bună autentificare și o comunicare clară, astfel încât să știe ce coduri să transmită înapoi la utilizator.

Componentele utilizate pentru transmiterea datelor către portul COM, dar și pentru o organizare din punctul de vedere al ordinii în care sunt primite datele de la mouse și transmise, au fost dezvoltate având la bază o descriere hardware realizată in VHDL.

O dezvoltare ulterioară a acestui proiect o poate reprezenta și în cazul în care utilizatorul nu folosește un mouse standard, deoarece dispune de un mouse care are mai multe funcționalități, fiind aplicată aceeași logică care a fost aplicată în proiectul curent.

Bibliografie

1. <https://www.fpga4student.com/2017/12/how-to-interface-mouse-with-FPGA.html>
2. <https://isdaman.com/alsos/hardware/mouse/ps2interface.htm>
3. <https://www.avrfreaks.net/sites/default/files/PS2%20Keyboard.pdf>
4. <https://www.digikey.com/eewiki/pages/viewpage.action?pageId=70189075>
5. <https://reference.digilentinc.com/reference/programmable-logic/basys-3/start>
6. <https://isdaman.com/alsos/hardware/mouse/ps2interface.htm>
7. <http://www.hardwarebook.info/AT_Keyboard/Mouse_protocol>