



بسمه تعالی

درس طراحی سیستم‌های نهفته مبتنی بر FPGA
آزمایش ۲: طراحی Audio Player با پردازنده NIOS II

پردیس دانشکده‌های فنی دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر
دکتر بیژن علیزاده

دستیاران آموزشی:

n.aghapour.s@gmail.com	نگار آقاپور
amirmahdijoudi@gmail.com	امیرمهدی جودی
sneginsafari@gmail.com	نگین سفاری

نیم‌سال اول ۱۴۰۱-۰۲

مدت آزمایش: سه جلسه

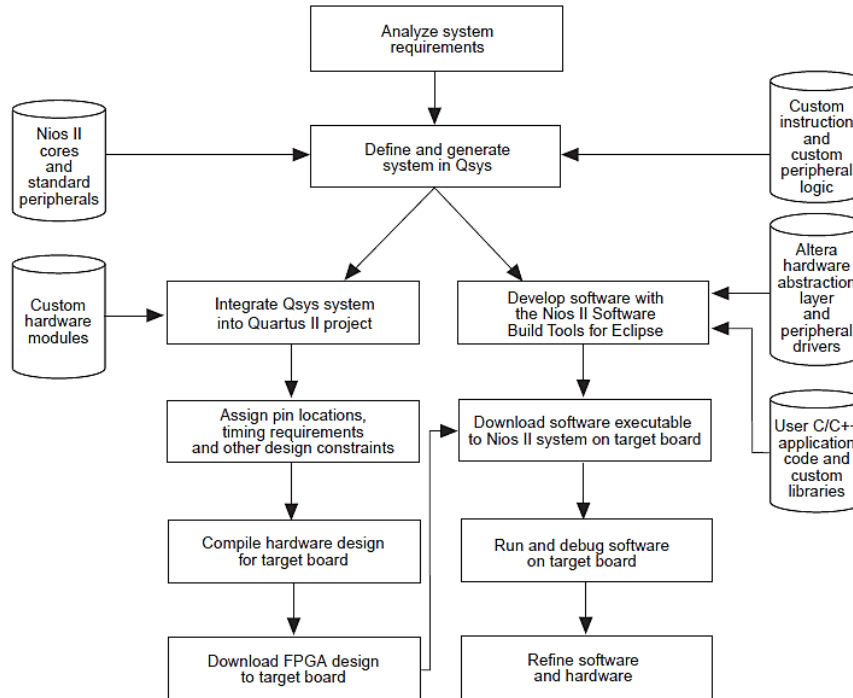
اهداف آزمایش:

- ✓ آشنایی با پردازنده NIOS II
- ✓ آشنایی با برنامه‌نویسی به زبان C برای پردازنده Nios II
- ✓ آشنایی با DE2 Media Computer
- ✓ آشنایی اولیه با روند طراحی سخت افزاری/نرم‌افزاری یک سیستم مبتنی بر Nios II

مقدمه

پردازنده Nios II که در سه نسخه سریع (f)، استاندارد (s) و اقتصادی (e) عرضه شده است، پردازنده‌ای ۳۲ بیتی و قابل پیکربندی^۱ با معماری RISC برای استفاده در FPGAها و SoCهای شرکت Altera است. البته امکان استفاده از این پردازنده به صورت ASIC نیز وجود دارد. یک سیستم کامل می‌تواند با قرارگیری چندین ماژول مختلف در کنار هسته پردازنده Nios II تشکیل شود. شکل ۱ روند کلی طراحی سخت‌افزاری/نرم‌افزاری یک سیستم مبتنی بر پردازنده‌ی Nios II را نشان می‌دهد. این روند را به صورت کامل در گزارش خود توضیح دهید.

^۱ Configurable



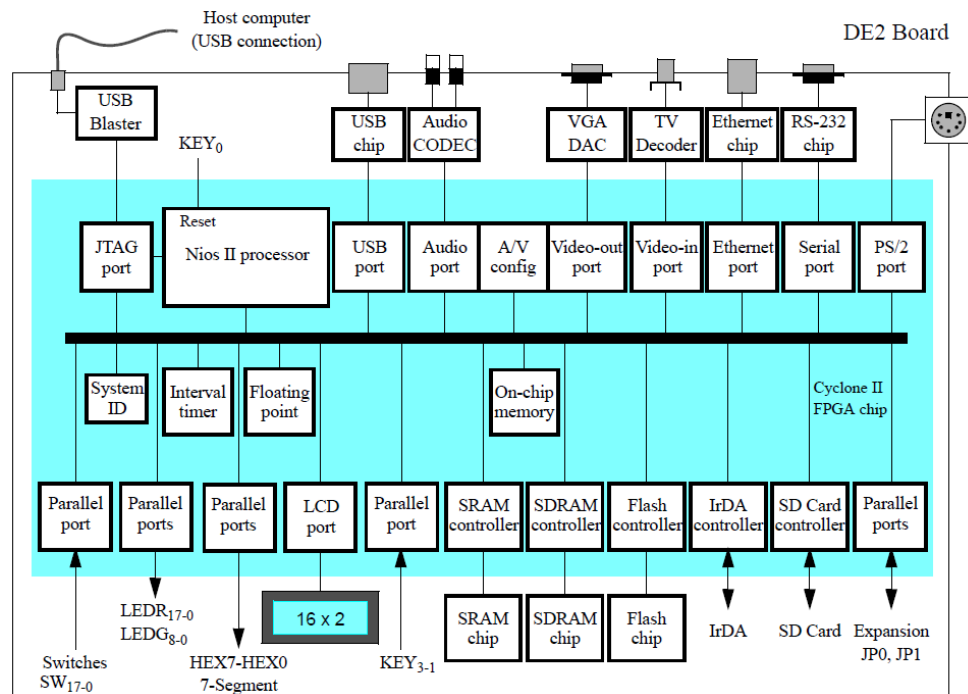
شکل ۱- روند طراحی سیستم مبتنی بر Nios II.

برای مثال شکل ۲ اجزای سیستم DE2 Media Computer را نشان می‌دهد که به کمک نرم‌افزار Qsys با متصل کردن IP Core های مختلف (از جمله خود هسته Nios II) به یکدیگر به کمک باس‌های استاندارد ساخته شده است. در این آزمایش به جنبه‌های نرم‌افزاری این سیستم خواهیم پرداخت و از این سیستم به عنوان بستر سخت‌افزاری آماده استفاده خواهیم کرد. برای انجام این آزمایش، آشنایی با زبان C ضروری است.

شرح آزمایش

در این آزمایش ابتدا یک سیستم سخت‌افزاری کامل از پیش طراحی شده به نام DE2 Media Computer را به کمک نرم‌افزار Quartus بر روی FPGA انتقال می‌دهیم و با یک برنامه آماده به کمک Nios II Software Build Tools for Eclipse آن را تست می‌کنیم (این کارها را می‌توان با Altera Monitor Program هم انجام داد اما جهت آشنایی با روند اصلی از دو برنامه دیگر استفاده می‌کنیم). سپس با مفهوم درایورنویسی آشنا می‌شویم و درایور ساده‌ای برای به کارگیری ماوس PS/2 در سیستم DE2 Media Computer می‌نویسیم. در نهایت یک دستگاه ضبط و پخش صدا با نمایش گرافیکی روی مانیتور و قابلیت ایجاد echo درست خواهیم کرد. به طور کلی آنچه در این آزمایش از شما خواسته می‌شود، پیاده‌سازی یک دستگاه ضبط صوت و پخش صدا با قابلیت اکو است که به صورت گرافیکی پیاده‌سازی می‌شود و به کمک ماوس، فرمان دستور موردنظر داده می‌شود. تمام این فرآیند

به صورت نرم‌افزاری و با نوشتن کد C انجام می‌شود. این بدان معناست که سخت افزار مورد نیاز ما آماده است و برای استفاده از آن، از توابع از پیش تعریف شده و یا دسترسی به آدرس رجیسترها استفاده می‌کنیم. در آزمایش-های بعد به کمک امکانات مختلف، بخشی از محاسبات زمان‌بر و سنگین به صورت سخت‌افزاری تسریع خواهند شد.



شکل ۲- بلوک دیاگرام DE2 Media Computer.

ابتدا Altera Monitor Program نسخه ۱۳,۰ را نصب نمایید تا IP Core ها، مستندات و خود سیستم‌های کامپیوتری نمونه از جمله DE2 Media Computer نصب شوند. سپس مراحل زیر را به ترتیب انجام دهید:

۱- آشنایی با DE2 Media Computer و اجرای کد نمونه

کد DE2 Media Computer یک کد نمونه برای برد DE2 می‌باشد که سخت‌افزار آن در شکل ۲ نمایش داده شده‌است. از مهم‌ترین بخش‌های این سخت‌افزار می‌توان به پردازنده Nios II اشاره کرد. کدهای برنامه‌های مختلف به زبان C می‌توانند روی این پردازنده اجرا شوند. یکی دیگر از رابط‌های مهم این کد، رابط parallel ports می‌باشد که به کمک آن می‌توان با ورودی و خروجی‌های دیجیتال کار کرد. برای مثال، راه اندازی یک 7-segment یا روشن و خاموش کردن یک LED روی برد DE2 نیازمند آن است که پین‌های مخصوص FPGA بتوانند به درستی

با اجزای روی برد ارتباط برقرار کنند و به عنوان خروجی عمل کنند یا برای switch های روی برد به صورت ورودی عمل کنند. برای ارتباط با یک مانیتور، یک راه استفاده از رابط VGA است که قابلیت ایجاد متن و تصویر را به صورت پیکسل فراهم می‌کند. کافی است تا با استفاده از بافرهای مخصوص متن یا شکل با بخش VGA مرتبط شد. ماوس و کیبوردهای قدیمی‌تر معمولاً از پروتکل PS/2 برای ارتباط با پردازنده استفاده می‌کردند. رابط PS/2 نیز در سخت‌افزار این آزمایش وجود دارد و برای کار با آن باید با این پروتکل آشنا شد. رابط صوتی برای پخش و ضبط صدا نیز جزو بخش‌هایی هست که در این آزمایش از آن استفاده می‌کنیم و برای کار با آن از بافرهایی برای ذخیره داده‌های صوتی استفاده می‌کنیم. روابط دیگری مثل مموری‌ها، اترنت، USB و نظایر اینها نیز در این نمونه وجود دارد. هم چنین قابلیت اضافه کردن بخش‌های دیگر به سیستم به کمک Qsys وجود دارد که در آزمایش‌های بعدی با آن آشنا می‌شوید.

برای استفاده از برخی روابط مثل PS/2 یا audio از وقفه^۲ استفاده می‌شود. لازم است کد وقفه به گونه‌ای باشد که تاثیر بدی در روند اجرای برنامه نداشته باشد.

برای آشنایی و درک بهتر موضوعات بالا مراحل زیر را طی کنید:

(۱) به مسیر زیر بروید:

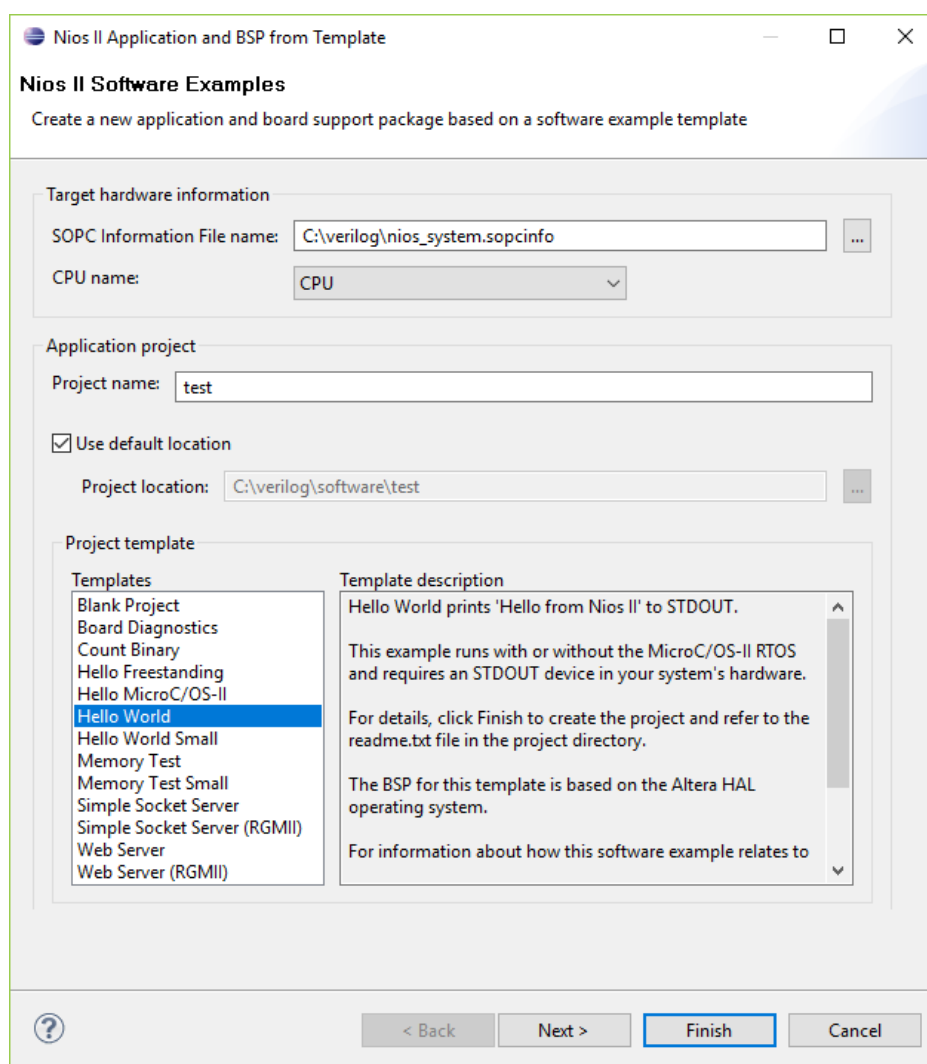
`altera\13.0spi\University_Program\NiosII_Computer_Systems\DE2\DE2_Media_Computer`
پوشه Verilog مربوط به DE2 Media Computer را به مسیر دیگری کپی کنید (حتماً این کار را انجام دهید و فایل‌های اصلی را تغییر ندهید زیرا برای آزمایش‌های بعدی استفاده خواهند شد). برد DE2 را به کامپیوتر متصل کنید و سیستم DE2 Media Computer را با پروگرام کردن FPGA با فایل `DE2_Media_Computer.sof` به آن منتقل کنید. در این آزمایش هیچ تغییری در سخت‌افزار DE2 Media Computer اعمال نخواهد شد. برای پروگرام کردن به منوی Tools رفته و از programmer استفاده کنید.

(۲) برنامه Nios II Software Build Tools for Eclipse را اجرا نمایید (این نرم‌افزار یک IDE بر مبنای Eclipse است که امکان کدنویسی برای پردازنده Nios II را فراهم می‌کند. نرم‌افزار به همراه Quartus II نصب شده است و از منوی Tools قابل دسترسی است). پس از تعیین یک workspace (که اطلاعات مربوط به پروژه‌های Eclipse و مکان قرارگیری آن‌ها را شامل خواهد شد) محیط Eclipse را مشاهده خواهید کرد.



(۳) از منوی File گزینه `New → Nios II Application and BSP from Template` را انتخاب نمایید و در پنجره ظاهر شده مسیر فایل `nios_system.sopcinfo` را که شامل اطلاعات IP Core های موجود در سیستم،

^۲ Interrupt

اتصالات بین آن‌ها و آدرس قرارگیری هر IP Core است، وارد نمایید (برنامه، آدرس‌هایی که شامل space هستند را برای آدرس فایل nios_system.sopcinfo قبول نمی‌کند). بر اساس این اطلاعات، درایورهای مورد نیاز (HAL) در قالب یک پروژه BSP (Board Support Package) ساخته خواهد شد. نام پروژه را به دلخواه انتخاب نمایید و در صورتی که می‌خواهید پروژه خود را در مسیر دیگری غیر از مسیر پیش‌فرض ایجاد کنید، آن را تغییر دهید. ابتدا مثال Hello World را انتخاب نمایید. شکل ۳، تنظیمات نمونه برای ایجاد پروژه جدید را نشان می‌دهد.



شکل ۳- صفحه ایجاد پروژه جدید در Eclipse.

۴) برای اجرای عادی برنامه بر روی پردازنده Nios II، از دکمه  و برای دیباگ آن از دکمه  استفاده کنید و سپس Nios II Hardware را انتخاب کنید. همچنین می‌توانید بر روی نام پروژه در پنجره Project

Explorer راست کلیک کرده و یکی از زیرمنوهای Run As... یا Debug As... را انتخاب نمایید. در این بخش برنامه Hello World را به صورت عادی اجرا نمایید (در بخش‌های بعدی استفاده از مد دیباگ و تعریف Breakpoint ها می‌تواند راهگشا باشد). با این کار کد از طریق رابط USB Blaster و رابط JTAG به حافظه‌های موجود در DE2 Media Computer انتقال می‌یابد و اجرا می‌شود. اولین اجرا به دلیل نیاز به کامپایل BSP که شامل تمامی درایورهای سیستم DE2 Media Computer است، کمی با تأخیر شروع خواهد شد اما در نهایت انتظار می‌رود با اجرای برنامه Hello World که کد ساده زیر است، در پنجره Console عبارت Hello from Nios II! چاپ شود.

```
#include <stdio.h>
int main()
{
    printf("Hello from Nios II!\n");
    return 0;
}
```

۵) حال یک مانیتور به پورت VGA برد وصل نمایید. کدهای موجود در مسیر زیر را (به غیر از فایل با پسوند ncf که مربوط به Altera Monitor Program است) به پروژه کپی کنید (فایل hello_world.c را پاک کنید) و برنامه جدید را اجرا کنید.

...\DE2_Media_Computer\app_software_HAL\media_interrupt_HAL

۶) کدهای مربوط به media_interrupt_HAL را به طور کامل مطالعه کنید و خلاصه عملکرد آن را گزارش نمایید. بدنه اصلی برنامه (تابع main) در فایل media_interrupt_HAL.c قرار دارد. در این کدها برای سرویس‌دهی به اجزای مختلف از جمله پورت PS/2 و پورت صدا از interrupt استفاده شده است و توابع ISR (Interrupt Service Routine) برای هر کدام در فایل جداگانه‌ای نوشته شده است. با مطالعه این کدها، مشخص کنید که در ابتدا بر روی مانیتور، چه عبارتی نمایش داده می‌شود؟ با فشردن کلیدهای ۱ و ۲ روی برد، چه عملیاتی اجرا می‌شود؟ این کدهای C از برنامه‌های آماده‌ی Altera Monitor Program هستند که در این آزمایش تغییرات مدنظر خود را در آن‌ها اعمال خواهیم کرد. از عملکرد صحیح کد اطمینان حاصل نمایید.

✓ خواسته‌ها:

- در گزارش خود حتما درباره‌ی ساختار alt_up_dev و peripheral های مختلف آن توضیح دهید.
- چرا برای تایمر از آدرس رجیستر base آن استفاده شده است؟ سایر رجیسترهای تایمر به چه منظور استفاده می‌شوند؟

- برای رجیستر کردن ISR ها از چه تابعی استفاده می‌شود؟ آرگومان‌های این تابع چه چیزهایی هستند؟
- همانطور که قبلاً گفته شد برای استفاده از PS/2 و Audio از وقفه استفاده می‌شود. توابع مربوط به مدیریت این وقفه‌ها را پیدا کرده و درباره‌ی نوع هر کدام از این وقفه‌ها (وقفه سخت افزاری یا نرم افزاری) توضیح دهید.
- درباره‌ی بافرهای صدا و نحوه‌ی مدیریت آن‌ها توضیح دهید.
- برای این قسمت حتماً کدهای اولیه را مطالعه کرده و عملکرد آن‌ها را توضیح دهید. سعی کنید که حتماً به سوالات بالا به طور کامل پاسخ دهید.

به منظور آشنایی بیشتر با DE2 Media Computer به فایل راهنمای این سیستم در پوشه doc در مسیر `altera\13.0sp1\University_Program\NiosII_Computer_Systems\DE2\DE2_Media_Computer` مراجعه کنید. در این مسیر فایل پروژه کامل این سیستم به دو زبان وریلاگ و VHDL قرار دارد. همچنین برنامه‌های نمونه به زبان C با و بدون استفاده از HAL^۳ در این مسیر قرار گرفته‌اند.

همچنین نسخه تغییر یافته‌ای از مستند DE2 Media Computer که فقط شامل بخش‌هایی است که خواندن آن‌ها برای انجام این آزمایش مفید است، در سایت درس بارگذاری شده است.

۲- راه‌اندازی درایور ماوس با خروجی PS/2

یک درایور از مجموعه‌ای از توابع نرم‌افزاری تشکیل شده است که به برنامه‌های کامپیوتری اجازه می‌دهد بدون این که از جزئیات پیچیده و سطح پایین یک وسیله خارجی آگاهی داشته باشند با آن وسیله خارجی ارتباط داشته باشند. از مهم‌ترین وسایل خارجی مورد استفاده می‌توان به ماوس و کیبورد اشاره کرد. برای اطلاع از وضعیت ماوس (موقعیت مکانی یا وضع کلیدها) باید با ماوس ارتباط داشته باشیم و بنابراین باید با درایور ماوس آشنا شوید. در این بخش هدف، استفاده از اطلاعات ارسالی توسط یک درایور برای ماوس با رابط PS/2 است. این یک پروتکل مخصوص ارسال و دریافت اطلاعات است که برای کار با آن لازم است تا با رجیسترهای آن در IP Core آن و چگونگی آدرس‌دهی آن آشنا شویم. به این منظور توضیحات زیر را به دقت مطالعه کنید.

^۳ Hardware Abstraction Layer

برای ارتباط با پورت PS/2 در DE2 Media Computer یک IP Core استفاده شده است. این هسته سخت‌افزاری، یک بافر FIFO ۲۵۶ بایتی دارد که اطلاعات را از PS/2 دریافت و ذخیره می‌کند. دسترسی به این بافر و نیز فعال کردن interruptها برای پورت PS/2 از طریق دو رجیستر memory-mapped انجام می‌شود. شکل ۴ این دو رجیستر را نشان می‌دهد.

Address	31	...	16	15	...	10	9	8	7	...	1	0		
0x10000100	RAVAIL			RVALID		Unused			Data				PS2_Data	
0x10000104							CE		RI				RE	PS2_Control

شکل ۴- رجیسترهای IP Core مورد استفاده برای پورت PS/2.

رجیستر PS2_Data قابلیت خواندن و نوشتن دارد. هنگامی که بیت ۱۵ (RVALID) برابر ۱ باشد، با خواندن این رجیستر اولین داده موجود در بافر در فیلد Data و تعداد دیتای موجود در بافر در فیلد RAVAIL خوانده می‌شود (این تعداد شامل دیتای خوانده شده است). همچنین با این کار RAVAIL یک عدد کاهش یافته و Data خوانده شده نیز از بافر FIFO داخلی حذف می‌شود. با نوشتن در فیلد Data از رجیستر PS2_Data می‌توان دستورات را به دستگاه PS/2 (که در اینجا ماوس است) انتقال داد.

رجیستر PS2_Control برای فعال کردن interruptها به کار می‌رود. زمانی که RE=1 باشد، پورت PS/2 یک درخواست وقفه را وقتی که RAVAIL بزرگتر از صفر باشد، تولید می‌کند. RI=1 زمانی اتفاق می‌افتد که درخواست وقفه در حال تعلیق است و با خالی کردن FIFO می‌توان این بیت را صفر کرد. بیت CE نشانگر رخداد خطا در هنگام ارسال دستور به دستگاه PS/2 است. اطلاعات بیشتر راجع به کنترلر PS/2 در مسیر زیر قرار دارد:

altera\13.0sp1\ip\University_Program\Input_Output\altera_up_avalon_ps2

دستوراتی که می‌توان به یک ماوس PS/2 ارسال کرد و جوابی که در ازای هر دستور دریافت می‌شود، در جدول ۱ آمده است. برای این که ماوس عملیات صحیحی انجام دهد، باید ابتدا ماوس را ریست کرد. پس از ارسال موفقیت آمیز دستور ریست (0xFF) انتظار می‌رود ماوس سه بایت متوالی 0xFAAA00 را بفرستد. بعد از دریافت پاسخ از ماوس می‌توان آن را با ارسال دستور 0xF4 فعال کرد و منتظر پاسخ 0xFA ماند (نحوه اجرای این کار در تابع مربوطه در فایل اصلی نوشته شده است). برای بررسی راه‌اندازی ماوس، می‌توانید پاسخ ماوس را در خروجی نمایش دهید. توجه داشته باشید که یک بار خواندن از بافر ماوس، باعث خالی شدن آن شده و داده جدید، جایگزین داده قبلی خواهد شد.

جدول ۱- لیست دستورات و پاسخهای مورد انتظار از ماوس.

Command Description	Command Byte	Response
Reset the mouse to default mode	0xFF	Responds with a 0xFA message, followed by a 2-byte message 0xAA00 if successful. A byte 0xFC will be sent otherwise to indicate an error.
Enable Mouse to send position and button status messages	0xF4	Responds with a single 0xFA byte if successful.
Disable Mouse	0xF5	Responds with a single 0xFA byte if successful. Send the Enable mouse command to resume receiving messages about the user's interactions with the mouse.

در حالت پیش فرض وقتی یک ماوس فعال می شود (برای مثال هنگام حرکت آن)، یک بسته (Packet) ۳ بایتی به پورت PS/2 می فرستد تا حالت خود را نشان دهد. فرمت بسته در جدول ۲ نشان داده شده است.

جدول ۲- اطلاعات ارسالی ماوس در قالب یک بسته ۳ بایتی.

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
byte 1	<i>y</i> overflow	<i>x</i> overflow	<i>y</i> sign	<i>x</i> sign	1	Middle btn	Right btn	Left btn
byte 2	Mouse <i>x</i> movement							
byte 3	Mouse <i>y</i> movement							

همانطور که مشاهده می شود تغییر در موقعیت ماوس با دو عدد ۹ بیتی مشخص می شود که یکی تغییرات در موقعیت های افقی و دیگری عمودی را مشخص می کند. اندازه حرکت در بایت های دوم و سوم و جهت حرکت در قالب بیت علامت (بیت های ۴ و ۵ از بایت اول) ارسال می شود. حرکت ماوس به سمت راست با تغییرات افقی مثبت و حرکت ماوس به سمت چپ با تغییرات افقی منفی بیان می شود. همچنین حرکت به سمت جلو و عقب نیز به ترتیب با تغییرات مثبت و منفی عمودی مشخص می شود. وضعیت سه دکمه ماوس با سه بیت پایین از بایت اول مشخص می شود که وقتی یک دکمه فشار داده شده باشد، بیت متناظر آن یک و در غیر این صورت صفر است. بیت شماره ۳ در بایت اول با یک شدن نشان می دهد که ۲ بایت بعدی متعلق به این بسته هستند. مکان اولیه ماوس پس از ریست شدن، در مختصات صفر و صفر که در اصل گوشه بالا سمت چپ تصویر هست و اشاره گر ماوس از آن نقطه تغییر مکان را با توجه به داده دریافت شده از ماوس شروع می کند. با مطالعه کد مربوط به ماوس، مشخص کنید که برای دریافت اطلاعات کامل مربوط به وضعیت فعلی ماوس، چند بار باید تابع PS2_ISR فراخوانی شود؟ فراخوانی این تابع به چه صورت خواهد بود؟

باید توجه داشت که پس از دریافت سیگنال interrupt، ماس به صورت بایتی داده هایی را به ترتیب ارسال می کند. به جهت آنکه موفقیت آمیز بودن ریست چک شود، لازم است که سه بایت داده ارسال شده به ترتیب در

تابع چک شوند. بعد از آنکه این عمل موفقیت آمیز بود، می‌توان با فرستادن دستور enable که در کد این تابع باید مدیریت شود، بعد از آن داده‌های متوالی ارسال شده را دریافت کرد.

اطلاعات کامل در مورد IP Core مربوط به ارتباط با پورت PS/2 در فایل راهنمای DE2 Media Computer قرار دارد.

برای پیاده‌سازی درایور ماوس PS/2 مراحل زیر را انجام دهید:

(۱) یک تابع C بنویسید که بر اساس بسته دریافتی حالت دکمه‌ها را بر روی LEDR[2..0] نمایش دهد. برای تغییرات در حرکت افقی (x) از HEX[5..4] استفاده کنید. وضعیت بیت سرریز بر روی HEX[6] و بیت علامت مربوط به حرکت افقی بر روی HEX[7] و به صورت ۰ یا ۱ نمایش دهید. تغییرات در حرکت عمودی (y) را روی HEX[1..0] نمایش دهید. وضعیت بیت سرریز را بر روی HEX[2] و بیت علامت مربوط به حرکت عمودی را بر روی HEX[3] نشان دهید. تغییرات لازم را در کد اولیه اعمال کنید. دسترسی به 7-SEG ها از طریق تابع HEX_PS2 و از این طریق سه بایت ارسال شده از ماوس PS/2 روی seven segment ها نمایش داده می‌شود.

(۲) در ادامه قابلیت درایور خود را به وسیله نگهداری مکان ماوس و محدود کردن مکانش به مرزهای مشخص افزایش می‌دهید. برای این کار شما باید ۵ متغیر تعریف کنید. دو متغیر برای نگهداری موقعیت افقی و عمودی، یکی برای نگهداری موقعیت دکمه ماوس و دو تا برای نگهداری ماکزیمم مختصات افقی و عمودی که ماوس می‌تواند در آن قرار بگیرد. توابع زیر را به کد اضافه کنید. تابع setMouseBounds(maxx,maxy) ماکزیمم مختصاتی که ماوس می‌تواند در آن قرار بگیرد را تعیین می‌کند و تابع getMouseState که موقعیت فعلی ماوس بر روی صفحه و وضعیت ۳ دکمه را برمی‌گرداند. این دو تابع را پیاده‌سازی کنید. همچنین مقدار ماکزیمم افقی و عمودی را به ترتیب روی ۳۱۹ و ۲۳۹ ست کنید.

✓ خواسته‌ها:

- دو تابع خواسته شده setMouseBounds و getMouseState گزارش شوند.
- نحوه راه‌اندازی درایور ماوس در نرم‌افزار DE2 Media Computer را توضیح دهید.

۳- طراحی Audio Player با نمایش گرافیکی و ایجاد Echo

در این بخش، سیستم کلی با نمایش گرافیکی نشان داده شده در شکل ۵، پیاده‌سازی و درستی‌سنجی خواهد شد. بدین منظور مراحل زیر را انجام دهید:

(۱) با استفاده از درایور ماوس خود، روی صفحه یک مکان‌نما (pointer) به صورت مربعی کوچک ایجاد کنید (استفاده از pixel-buffer). در صورت استفاده از pointer به صورت فلش نمره امتیازی تعلق می‌گیرد (توضیح در ادامه).

(۲) روی صفحه نمایش با استفاده از تابع `alt_up_pixel_buffer_dma_draw_box` مربع‌هایی به صورت دکمه برای ضبط، پخش معمولی و پخش همراه با echo ایجاد کنید. توجه نمایید که کد اصلی قابلیت ضبط و پخش صدا با استفاده از کلیدهای دوم و سوم را دارد. این امکان را از کد حذف نکنید. برای انتخاب رنگ می‌توانید از جدول ۳ استفاده کنید.

جدول ۳- اطلاعات رنگ مربوط به پیکسل‌ها.

0 ... 4 5 ... 10 11 ... 15

Blue	Green	red
------	-------	-----

(۳) یک تابع بنویسید که در صورت کلیک روی هر یک از این دکمه‌ها با توجه به محدوده آن در صفحه نمایشگر، به کمک محتوای `l_buff` و `r_buff` و تابع `alt_up_audio_play_1` این عملیات را انجام دهد. توجه داشته باشید که باید به کمک شمارنده‌ای تعداد داده‌های موجود در بافرها که برای پخش فرستاده شده‌اند را داشته باشیم تا بتوانید این روند را کنترل کنید.

(۴) به منظور echo کردن باید برخی نمونه‌های صدای قبلی با نمونه کنونی جمع شوند. بدین جهت تابعی بنویسید که نمونه‌های `x[n-1000]` و `x[n-5000]` را با صدای خروجی جمع کند و سپس آن را پخش کند. توجه داشته‌باشید که این بخش از کد روند مشابهی با بخش قبل دارد و فقط در داده فرستاده شده برای پخش متفاوت است.

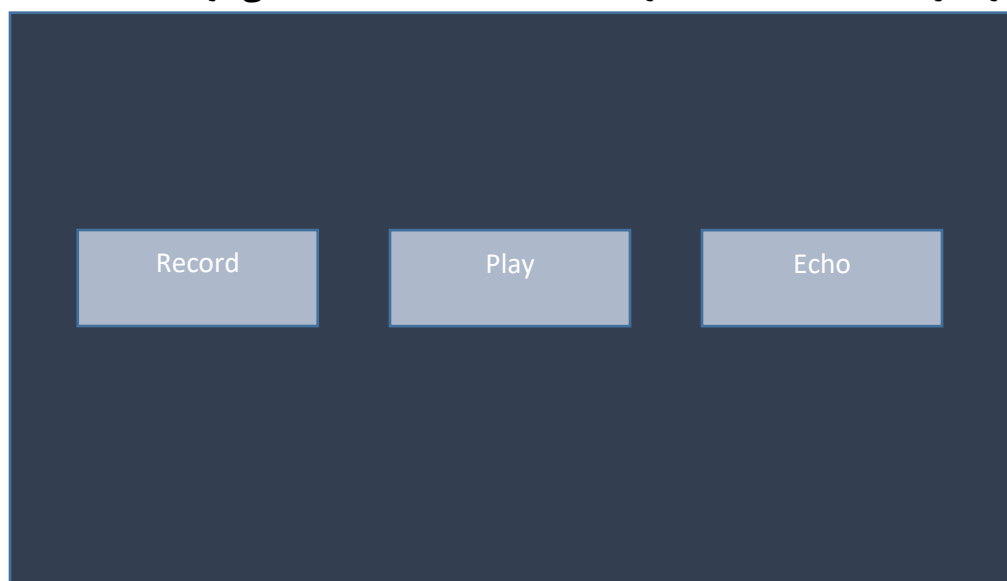
(۵) برنامه خود را روی برد اجرا کنید و صحت راه‌اندازی صحیح ماوس و عملیات مربوط به ضبط و پخش صدا را بررسی نمایید.

برای دیباگ (عیب‌یابی) عملکرد کد ریکورد، پخش و اکو می‌توانید از ترمینال برای اجرای دستورات استفاده کنید.

بخش مربوط به Pixel Buffer DMA Controller را از فایل راهنمای IP Core های ویدئو در پوشه doc در مسیر زیر مطالعه نمایید. برای این آزمایش خصوصاً فهم دقیق عملکرد Back Buffer اهمیت دارد.

altera\13.0sp1\ip\University_Program\Audio_Video\Video

با مطالعه‌ی راهنمای DE2 Media Computer و نیز کدهای نمونه با نحوه کار با ورودی/خروجی آنالوگ آشنا شوید. تابع alt_up_pixel_buffer_dma_draw_box را مطالعه کنید تا با نحوه نمایش روی صفحه آشنا شوید. همچنین با نحوه کار با VGA Pixel Buffer و VGA Character Buffer آشنا می‌شوید.



شکل ۵- نمونه‌ای از نمایش گرافیکی صفحه.

بخش امتیازی

برای تعریف شکل ماوس یک ماتریس ۱۶ در ۸ ایجاد کنید. هر درایه ماتریس دارای یکی از این سه مقدار است:

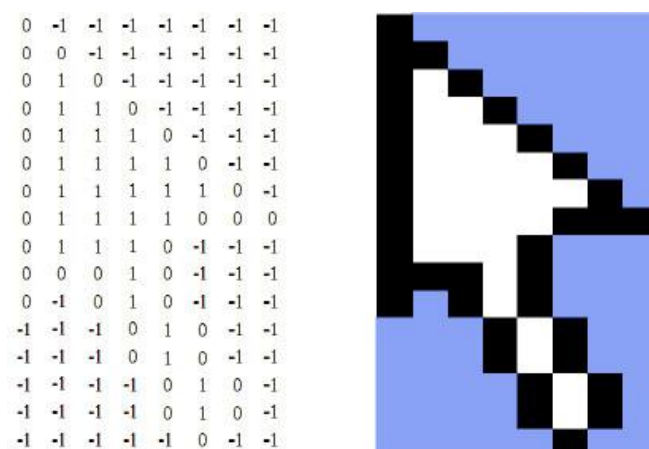
۰: نشان می‌دهد پیکسل متناظر اشاره‌گر ماوس مشکی است.

۱: نشان می‌دهد پیکسل متناظر اشاره‌گر ماوس سفید است.

۱- : نشان می‌دهد پیکسل متناظر اشاره‌گر ماوس شفاف است. این پیکسل‌ها هنگام حرکت ماوس رسم

نمی‌شوند.

برای مثال شکل ۶ یک نمونه از اشاره گر ماوس و ماتریس متناظرش را نشان می‌دهد. رنگ آبی همان پیکسل‌های شفاف هستند.



شکل ۶- ماتریس و تصویر اشاره گر ماوس.

✓ خواسته‌ها:

- توابع مربوط به نمایش ماوس، شروع عملیات ضبط و پخش و echo گزارش شوند.

نکات مهم:

- ۱) در این آزمایش عملکرد صحیح سیستم اهمیت زیادی دارد. به عنوان مثال کیفیت صدا باید مطلوب باشد. تشخیص مشکلات، تحلیل آن‌ها و حل مشکلات سیستم بخشی از این آزمایش است.
- ۲) نیاز است هر آنچه را که در بخش ۱ خواسته شده است به دقت مطالعه نمایید. این کار بخشی از انجام آزمایش دوم است و امکان راهنمایی در این زمینه وجود ندارد. در مورد کد، خواندن کامنت‌های موجود در کد می‌تواند کمک شایانی در فهم دقیق آن کند. برای انجام این آزمایش که کاملاً نرم‌افزاری است، لازم است کاربرد `interval_timer`، `VGA_pixel_buffer`، و همچنین تمامی ISRهای موجود را کاملاً تحلیل نمایید.
- ۳) به طور کلی ارتباط Nios با IP Core ها با خواندن و تغییر رجیسترهای آن‌ها (در آدرس مشخص خودشان) امکان پذیر است (همچنین ارتباط از طریق وقفه و نیز DMA ها، دیگر مجموعه ابزارهای ارتباطی بین Nios و سایر IP Core ها را تشکیل می‌دهند). درایورها شامل روال‌هایی سطح بالا از این

ارتباطات هستند. به عنوان مثال در کد مربوط به بخش دوم ارتباط با interval_timer را (که در bsp درایور مناسبی برایش تعریف نشده است) با سایر IP Core ها مقایسه نمایید.

(۴) لزومی به تعریف توابع مطابق متن آزمایش وجود ندارد اما نتایج خواسته شده باید به دست آید. روند این آزمایش بدین صورت است: به هر طریقی که ممکن است، مکان ماوس را تعریف و به صفحه نمایش (طول ۰ تا ۳۱۹ و عرض ۰ تا ۲۳۹) محدود کنید و آن را روی HEX Display نشان دهید به طوری که با حرکت ماوس مکان آن نشان داده شود. سپس ماوس را به صورت یک مربع (استفاده از شکل فلش امتیازی است) در مکان مد نظر رسم نمایید (باید با حرکت ماوس شکل مکان قبلی آن بازگردد). دکمه های ضبط، پخش و پخش صدا همراه با اکو را ایجاد کنید و عملکردهای مربوطه را به آن ها اختصاص دهید (با کلیک روی آن ها و یا دریافت دستور سریال از کاربر، عملکرد مربوطه اجرا شود). توجه نمایید که دو عملکرد ضبط و پخش صدا در کد اولیه پیاده سازی شده است.

(۵) کد ISR ها را محدود و کم کنید. سعی کنید ارتباط بین ISR با برنامه اصلی را با متغیرهای مشترک کنترل کنید و از انتقال برنامه های زمانبر به داخل ISR ها پرهیز کنید.

(۶) پیروی از قالب خاصی در گزارش مد نظر نیست، اما ترجیحاً می توانید از قالب ارائه شده برای تکالیف کامپیوتری استفاده نمایید.

(۷) آپلود کدهای C به همراه فایل گزارش ضروری است. لازم است کدهای بخش های ۱، ۲ و ۳ را در پوشه های جداگانه آپلود کنید (همه را در داخل یک فایل zip. قرار دهید).

موفق و سلامت باشید.