به نام خدا

پروژه درس مدار واسط

طراحی قفل سخت افزاری برای پورت های سریال، موازی و USB

زیر نظر: استاد پژمان خدیوی

تهیه کننده:

حسام احمدی

دی ماه 1387

فهرست مطالب

به نام خدا

فصل اول: آشنایی با قفل سخت افزاری[1]، تعاریف و کاربردها

قفل سخت افزاری چیست؟

به منظور آشنایی با قفل سخت افزاری، مروری بر سایت های مختلف اینترنتی انجام شده است. به منظور جستجو در سایت های انگلیسی زبان، از کلمه "dongle" استفاده شده است. بنابر اظهارات سایت wiseGreek[2] نام dongle برای قفل سخت افزاری نام مناسب و نا معقول و البته رایج است.

منشا اصلی قفل سخت افزاری دقیقا مشخص نیست. دیکشنری "American Heritage" در این باره می گوید: "شاید این یک نو آوری مطلق بوده است."[3]

درادامه به ذکر 2 مورد از تعاریف موجود برای dongle می پردازیم. این تعاریف به زبان اصلی آورده شده اند.

A security or copy protection device for commercial microcomputer programs. Any electronic key or transferable ID required for a program to functions.[4]

This term refers to a small hardware key that plugs into the serial or parallel port of a computer. It is used to ensure that only authorized users can copy or use certain software .they're only used with ultra-expensive, high end software programs that most people have never heard of, where the high-priced program runs, it checks the dongle verification before continuing. If it doesn't find the dongle, the program usually quits.[5]

بنابر تعریف بیان شده چنانچه از یک سخت افزار خاص برای قفل گذاری روی یک نرم افزار استفاده شود، به آن قفل سخت افزاری گوییم. این روش قفل گذاری به گونه ای است که بدون وجود قفل، برنامه مورد نظر در حالت نمایشی کار می کند یا اصلا اجرا نمی شود.

در سال 1980 برنامه ای به نام "wordcraft" برای اولین بار به صورت امروزی از قفل سخت افزاری برای حفاظت از خود استفاده کرد.

---

[1] Security Dongle
[2] www.wisegreek.com/what_is_a_dongle.html

[3] www.wikipedia.com
[4] www.theidm.com/index.cfm
[5] www.mavidesigne.com/glossary.html

نحوه عملکرد برخی از قفل های سخت افزاری

در ابتدا قفل های سخت افزاری مدارهای پسیو ساده ای بودند که داده ها را طبق الگوی از قبل تعیین شده ای، به پورت سیستم ارسال می کردند. بزرگترین مشکلی که ابتدا به چشم می خورد این بود که پورت کامپیوتر توسط قفل سخت افزاری اشغال شده و کاربرد اصلی خود را از دست می داد.

دسته دیگر از قفل های سخت افزاری دارای حافظه ای درحد چند بایت و ساختاری ساده بودند. تولید کننده نرم افزار یک یا چند بایت داده را در قفل می نوشت. برنامه در حال اجرا این اطلاعات را چک کرده و در صورتی که قفل مربوط در پورت قرار داشت، برنامه به کار خود ادامه می داد.

دسته دیگری از قفل ها دارای ساختار پیچیده تر و حافظه ای در حد چند کیلوبایت بودند. تولید کننده نرم افزار بخش کوچکی از کد های برنامه را در این حافظه قرار می داد. بنابراین در صورت عدم وجود این قفل، برنامه اصلا اجرا نمی شد.

بعدا قفل ها به دستگاه های اکتیو که شامل UART و حتی میکروکنترلر بودند، تبدیل شدند. در این نوع قفل ها تراکنش های خاصی با PC انجام می شود. هم اکنون انجام این تراکنش ها با استفاده از قفل های سخت افزاری متصل شونده به پورت USB در حال رایج شدن و نمونه های قدیمی تر که به پورت های سریال و موازی وصل می شدند در حال منسوخ شدن می باشند.

علاوه بر موارد بالا، بیشمار الگوریتم دیگر برای تامین امنیت نرم افزار های کامپیوتری قابل پیاده سازی است که در این پروژه به یکی از انواع آن ها می پردازیم.

ملاحظات کلی برای طراحی قفل های سخت افزاری

ضعف های بالقوه ای در پیاده سازی نرم افزار و قفل مرتبط با آن وجود دارد. به عبارت دیگر تدابیر زیادی لازم است تا قفل سخت افزاری طراحی شده قابل شکستن یا کپی برداری نباشد. قفل های سخت افزاری پیشرفته دارای روش های مخفی سازی و رمز نگاری قوی هستند. به علاوه در ساخت این قفل ها باید از تکنیک هایی استفاده کرد تا امکان انجام مهندسی معکوس وجود نداشته باشد. عدم توجه به این موارد موجب می شود که فرد قفل شکن با استفاده از device driver بتواند از قفل کپی برداری کند.

امروزه طراحان قفل های سخت افزاری از کارت های هوشمند در طرح های خود اقتباس می کنند. دراین بین کارت های بانکی بیشتر مورد توجه بوده است.

طراحی قفل سخت افزاری در واقع تعیین نحوه ارسال داده توسط نرم افزار محافظت شده به قفل سخت افراری و چگونگی پردازش و ارسال آن توسط قفل است.
یک قفل سخت افزاری خوب علاوه بر اینکه باید از نرم افزار در مقابل استفاده های غیر قانونی محافظت کند، باید خود در مقابل hack شدن و کپی برداری محافظت شده باشد. این ویژگی زمانی بدست می آید که قفل سخت افزاری مانند یک تابع مجهول با دامنه و برد حداکثری عمل کند زیرا در صورتی که همواره یک رشته خاص از داده ها به قفل ارسال شود و قفل نیز رشته هایی از داده های قابل پیش بینی را بازگرداند، از نظر امنیتی قفل خوبی به شمار نمی آید و به راحتی قابل کپی برداری است.
به طور کلی ویژگی های یک قفل خوب عبارتند از:

1- قابلیت وصل شدن به یکی از پورت های معمول
2- بدون نیاز به تغذیه خارجی. به عبارت دیگر تغذیه قفل از خود پورت تامین شود
3- پورت را به طور کامل اشغال نکند و امکان استفاده از پورت توسط سایر دستگاه های دیگر وجود داشته باشد
4- در مقابل مهندسی معکوس و کپی برداری حفاظت شده باشد
5- دارای الگوریتمی باشد که حدس زدن آن از نظر زمانی برای قفل شکن به صرفه نباشد[6]

در این نوشته به طراحی قفل سخت افزاری برای پورت های SPP[7] ، Serial و USB[8] می پردازیم. در این طراحی ها سعی شده کلیه ویژگی های ذکر شده در بالا وجود داشته باشند.

[6] Programming and customizing the avr microcontrollers by DHANAJAY V.GADRE
[7] Simple Parallel Port
[8] Universal Serial Bus

فصل دوم: طراحی قفل سخت افزاری برای SPP

الگوریتم های بیشماری برای طراحی قفل های سخت افزاری امکان پذیر است. قفل های سخت افزاری اولیه از روش وصل کردن بعضی از ورودی های پورت پرینتر به خروجی های آن استفاده می کردند. در این صورت با وجود قفل سخت افزاری، نرم افزار با تولید خروجی در پورت انتظار دریافت همان داده های ارسال شده را داشت و به این صورت حضور قفل سخت افزاری مناسب را آزمایش می کرد. اگر چه این پیاده سازی این الگوریتم بسیار ساده است اما به راحتی قابل تشخیص و کپی برداری می باشد.

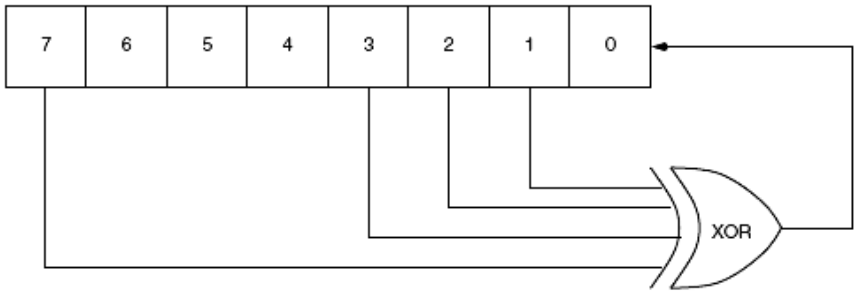
الگوریتم هایی که در این پروژه به کار خواهد رفت به گونه ای است که قفل داری رفتار تناوبی[9] بسیار کندی است. به عبارت دیگر امکان چک کردن ورودی و خروجی های قفل وجود نداشته یا بسیار وقت گیر و عملا ناممکن است. برای ایجاد یک رفتار تناوبی کند، از یک تابع منطقی استفاده شده است که ورودی آن توسط PC تامین می شود و خروجی آن مجددا به PC ارسال می شود.

ایده استفاده شده برای تولید خروجی توسط قفل، مشابه ایده ای است که برای تولید اعداد شبه تصادفی[10] استفاده می شود. برای تولید چنین اعدادی از یک LFSR[11] استفاده می شود. قفل سخت افزاری مبتنی بر LFSR دارای سیکل تناوبی بسیار طولانی می باشد و حدس زدن آن بسیار مشکل است. به علاوه با تغییرات جزئی در برنامه قفل می توان به شدت پیچیدگی قفل افزود.از ویژگی های مهم LFSR آن است که اگر مسیرهای فیدبک آن به درستی انتخاب شوند، سیکل تناوب آن به $2^{n-1}-1$ می رسد که n تعداد بیت های shift register آن است. برای استفاده از LFSR باید یک مقدار اولیه یا seed در آن بارگذاری نمود در این صورت به ازای هر یک شیفت داده های موجود در shift register، یک عدد تصادفی تولید می شود[12].

در شکل زیر یک LFSR 8 بیتی با feedback tap هایی روی بیت های 1، 2، 3 و 7 نشان داده شده است.همچنین برای نمونه چند خروجی آن به ازای seed=1 آورده شده است.

[9] Repeat cycle
[10] Pseudorandom
[11] Linear Feedback Shift Register
[12] Programming and customizing the avr microcontrollers by DHANAJAY V.GADRE

```
Seed = 1
2  5  b  16 2c 58 b1 63 c7 8f 1e 3d 7a f4 e8 d0
a1 43 87 f  1f 3f 7f ff fe fc f9 f2 e4 c8 90 21
42 85 a  14 29 53 a7 4f 9f 3e 7d fa f5 ea d5 aa
55 ab 57 ae 5c b8 70 e0 c1 83 6  c  18 31 62 c5
8a 15 2b 56 ac 59 b3 66 cc 99 32 65 cb 97 2f 5f
bf 7e fd fb f7 ef de bc 79 f3 e6 cd 9b 37 6e dd
bb 77 ee dc b9 72 e5 ca 95 2a 54 a9 52 a5 4a 94
28 51 a2 44 89 12 25 4b 96 2d 5a b4 68 d1 a3 46
8c 19 33 67 ce 9c 39 73 e7 cf 9e 3c 78 f1 e3 c6
8d 1b 36 6c d8 b0 61 c2 84 8  11 22 45 8b 17 2e
5d ba 75 eb d7 af 5e bd 7b f6 ed db b7 6f df be
7c f8 f0 e1 c3 86 d  1a 34 69 d3 a6 4d 9a 35 6b
d6 ad 5b b6 6d da b5 6a d4 a8 50 a0 41 82 4  9
13 27 4e 9d 3b 76 ec d9 b2 64 c9 92 24 49 93 26
4c 98 30 60 c0 81 3  7  e  1d 3a 74 e9 d2 a4 48
91 23 47 8e 1c 38 71 e2 c4 88 10 20 40 80 1
```



8 بیتی با feedback tap هایی روی بیت های 1، 2، 3 و 7

جدول زیر برای طراحی LFSR به کار می رود. ستون اول تعداد بیت های shift register و ستون آخر محل feedback tap ها را مشخص می کند. هر چند محل های دیگری برای feedback tap ها وجود دارد اما برای دستیابی به بزرگترین سیکل تناوبی باید مطابق جدول انتخاب شوند. ستون وسط سیکل تناوب LFSR را مشخص می کند. مشاهده می شود که برای یک shift register با 20 بیت این سیکل به بیش از یک میلیون می رسد.

BITS	SEQUENCE LENGTH	TAPS
9	511	3,8
10	1023	2,9
11	2047	1,10
12	4095	0,3,5,11
13	8191	0,2,3,12
14	16,383	0,2,4,13
15	32,767	0,14
16	65535	1,2,4,15
17	131,071	2,16
18	262,143	6,17
19	524,287	0,1,4,18
20	1,048,575	2,19

در ادامه به پیاده سازی قفل سخت افزاری بر اساس استفاده از LFSR می پردازیم. در کلیه این طرح ها برای استفاده از LFSR ، با سازوکار مشخصی یک مقدار اولیه توسط PC به قفل فرستاده می شود و سپس بر اساس این مقدار اولیه، چندین عدد شبه تصادفی توسط PC و قفل سخت افزاری ساخته می شوند. سپس قفل سخت افزاری مقادیر تولید شده را به PC ارسال می کند. صحت قفل سخت افزاری با مقایسه مقادیر دریافت شده از قفل و مقادیر مورد انتظار، توسط نرم افزار حفاظت شده بررسی می شود.

در این طراحی از یک میکروکنترلر خانواده AVR به شماره AT90s2343 استفاده شده است. این میکروکنترلر دریافت و ارسال داده بین PC و قفل را بر عهده دارد. به علاوه LFSR مورد نیاز نیز توسط این میکروکنترلر به صورت نرم افزاری پیاده سازی شده است.

این میکروکنترلر دارای ویژگی های زیر است[13]:

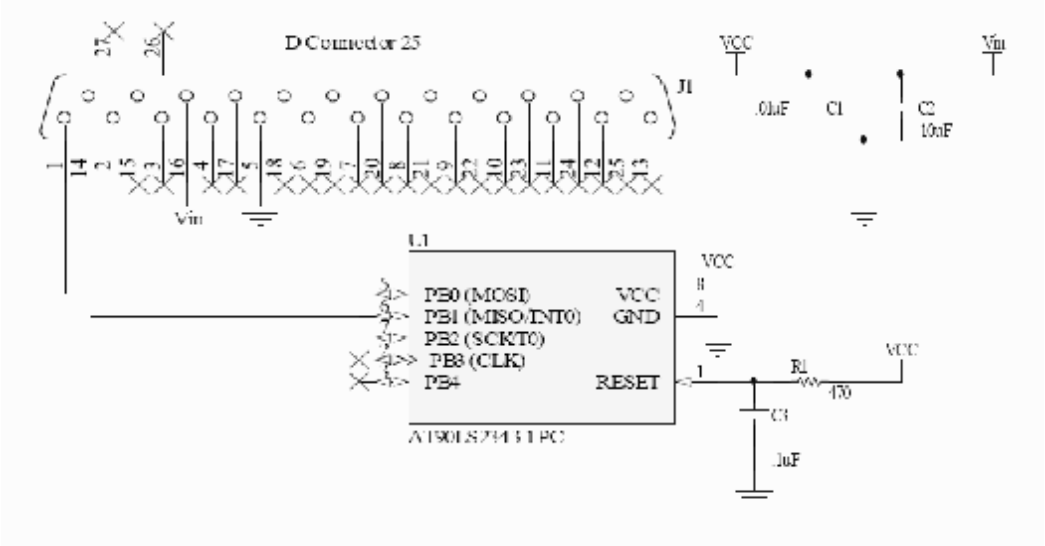
- معماری RISC
- 23*8 رجیستر همه منظوره
- 2KB حافظه flash
- 128 بایت RAM
- 128 بایت EEPROM
- امکان Program Lock برای flash program و EEPROM
- تایمر/شمارنده 8 بیتی
- Watchdog timer
- امکان ISP
- وقفه داخلی و خارجی
- پنج خط I/O برنامه پذیر
- ولتاژ کاری 4 تا 6 ولت

وجود ویژگی هایی از قبیل اندازه کوچک، امکان قفل کردن محتویات flash و EEPROM، وقفه خارجی و... باعث می شود که این میکروکنترلر گزینه مناسبی برای این کار باشد.

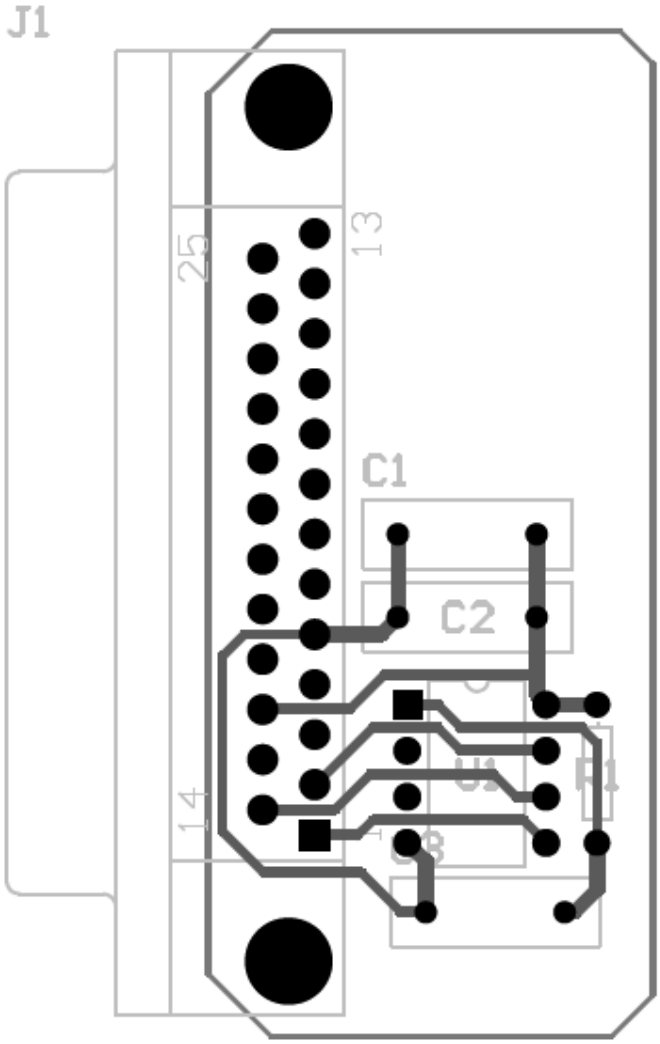
شماتیک مدار مورد استفاده برای این قفل در زیر آمده است. همانطور که مشاهده می شود از خط 16 برای تامین تغذیه مدار قفل استفاده شده است و خطوط 1،2 و 14 در عملیات I/O استفاده شده اند.

از یک مدار RC نیز برای ریست خودکار میکرو بعد از وصل تغذیه استفاده شده است.

[13] www.atmel.com



شماتیگ قفل سخت افزاری برای SPP



PCB مربوط یه قفل سخت افزاری برای SPP

نرم افزارمیکروکنترلر

میکروکنترلر طوری برنامه ریزی شده که به وقفه خارجی پاسخ دهد. پس از فعال شـده خـط 16 پورت توسط نرم افزار محافظت شده، تغذیه میکروکنترلر وصل می شود و بعد از مدت کوتاهی ریست و آماده به کار می شود. میکروکنترلر آمادگی خود را با یک کردن بیـت دوم از پورت خود به PC اعلام می کند. بنابراین نرم افزار محافظت شده باید قبل از ارسال Seed به میکروکنترلر از یک بودن این خروجی اطمینان حاصل کند.

برای ارسال Seed از PC به قفل، کامپیوتر بیت های Seed را یکی یکـی روی پـین 1 از پـورت SPP قرار می دهد و توسط پین شماره 2 پورت SPP یک تقاضای وقفه خـارجی ارسـال مـی کند. در سرویس روتین وقفه خارجی زیر برنامه ای وجود دارد که داده را از پایه صفر پورت میکرو(یا همان پین 1 از پورت SPP) بر می دارد و در محل مناسبی ذخیره می کند. به محض آنکه از این طریق تعداد بیت 8 بیت داده از PC دریافت شود میکروکنترلر وارد فاز بعدی می شود. همانطور که مشاهده می شود، ارسال بیت های مربوط به seed می تواند پشـت سـر هم و یا در فواصل زمانی مشخصی انجام نشود. این ویژگی باعث می شود که تشخیص نحـوه عملکرد قفل توسط قفل شکن بسیار مشکل شود. به علاوه تعداد بیت های مربوط به seed بـا تغییرات نرم افزاری به راحتی قابل افزایش است و می توان پیچیدگی نحـوه عملکـرد قفـل را افزایش داد.

پس از دریافت کامل seed، زیر برنامه ای به نام TRANS_MAN در میکروکنترلر اجرا مـی شود. این زیر برنامه موجب تولید 5 عدد شبه تصادفی شده و آن را به PC ارسال مـی کنـد. تعداد این اعداد تولید و ارسال شده بدون محدودیت خاصی قابل افزایش است و فقط موجب کند شدن عملیات چک کردن قفل توسط نرم افزار محافظت شده خواهد شد.

نحوه ارسال اطلاعات به PC با دریافت آن از PC متفاوت انتخاب شده است. دراین مرحلـه از روش Destination initiate strobe استفاده می شود. PC با بالا بـردن بیـت صـفرم پـورت میکروکنترلر، آمادگی خود را برای دریافت اطلاعات از میکروکنترلر اعلام مـی کنـد و پـس از آن میکروکنترلر بیت مورد نظر را روی پایه دوم از پورت خود قرار مـی دهـد.در اینجـا نیـز ارسال اطلاعات می تواند در فواصل نـا مـنظم انجـام شـود و موجـب ایجـاد اخـتلال در رونـد تشخیص نحوه عملکرد قفل توسط قفل شکن شود.

کدهای برنامه میکروکنترلر به زبان اسمبلی در زیر آورده شده است.

```
.include "2343def.inc"

.org $000
rjmp MAIN ; Reset Handler
rjmp EXT_INT0 ; IRQ0 Handler
```

```
MAIN:
        ldi r22, $08
        ldi r16, low(RAMEND) ; Main program start
        out SPL, r16
        ldi r16, $04 ; PB2 as output and other as input
        out ddrb, r16
        ldi r16, $03
        out portb, r16
        ldi r16, $40
        out gimsk,r16
        ldi r16, $02
        out mcucr,r16
        sei

READY:
        sbi portb,2
        rjmp READY

CODE_GEN: ; It is a LFSR
        clt
        ldi r17, $04
        and r17, r16

        sbrs r17,2
        inc r18

        ldi r17, $10
        and r17, r16

        sbrs r17,4
        inc r18

        ldi r17, $40
        and r17, r16

        sbrs r17,6
        inc r18

        cpi r18,1
        clc
        brne NEXT
        sec

NEXT:
        rol r16
        ret
```

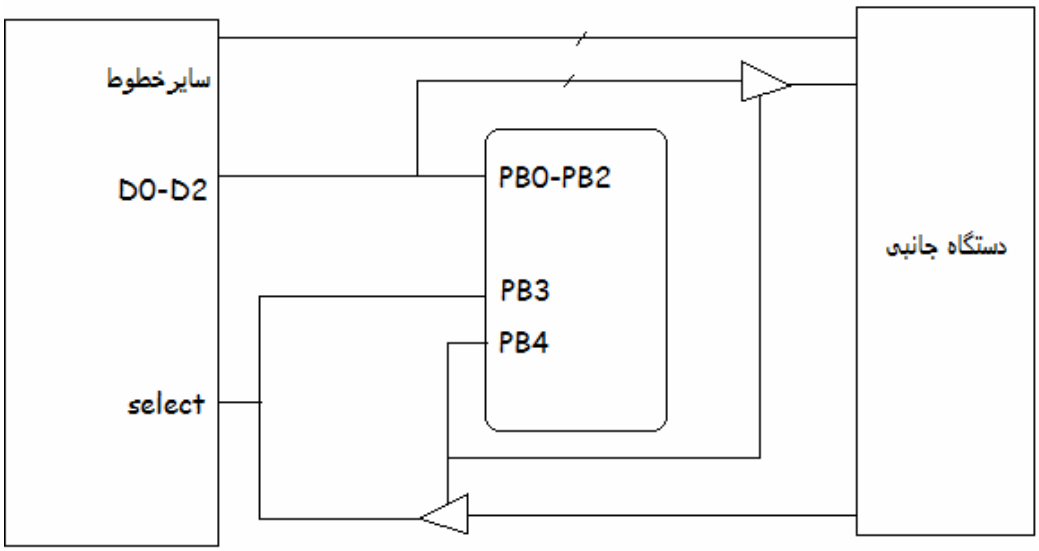
```
SEND: ; it sends data to PC
        mov r20,r16
        ldi r21,$08
again:
        sbis portb,0
        rjmp again
        nop
        sbrc r20,7
        sbi portb,2
        rol r20
        dec r21
        tst r21
        brne again
        ret


EXT_INT0: ; for receiving one bit from PC
        sbic portb,0
        sbr r16,0
        rol r16
        dec r22
        tst r22
        brne next_inbit
        rjmp TRANS_MANAGER ; if receives 8 bits from PC
next_inbit:
        ldi r23,$40
        out gifr, r23 ; Clear interrupt flag
        sei
        reti


TRANS_MANAGER:
        ldi r22,$05 ; Generate 5 code and send them to PC
next_data:
        rcall CODE_GEN
        rcall SEND
        dec r22
        tst r22
        brne next_data
        ldi r22,$40
        out gifr, r22 ; Clear interrupt flag
        ldi r22,$08
        sei
        ret
```

فصل سوم: طراحی قفل سخت افزاری برای SPP با قابلیت pass-through

همانطوری که از در شماتیک قفل سخت افزاری قبل مشاهده شد، این قفل به طور کامل پورت SPP را اشغال می کند و امکان اتصال سایر وسایل جانبی مثل پرینتر به این قفل وجود ندارد. در ادامه می خواهیم به ارتقا این قفل سخت افزاری بپردازیم به طوریکه با وجود اتصال قفل به پورت کامپیوتر امکان اتصال یک وسیله جانبی دیگر به پورت وجود داشته باشد. این کار pass-through parallel port نامیده می شود.

برای رسیدن به این هدف از روش پایش سیگنال[14] استفاده می کنیم. در این روش قفل سخت افزاری دائما سیگنال های خروجی خاصی از پورت کامپیوتر را پایش می کند و به محض مشاهده سیگنال های خاصی در خروجی، بدست گرفته شده کنترل پورت توسط نرم افزار محافظت شده را تشخیص می دهد و آماده دریافت، پردازش و ارسال داده می شود. این روش شبیه آدرس دهی دستگاه های مختلفی است که به یک باس آدرس وصل شده اند.

همانطوری که مشاهده می شود این روش مستلزم آن است که قفل سخت افزاری همواره در مسیر برخی از سیگنال های پورت باشد. برای جلوگیری از ایجاد وقفه در مسیر سیگنال ها، قفل باید به طور موازی با دستگاه جانبی دیگر قرار گیرد. همچنین به منظور جلوگیری از تداخل سیگنال ها باید ملاحظاتی در نظر گرفته شود. یکی از ساده ترین روش ها برای جلوگیری از تداخل سیگنال ها، استفاده از بافر های 3 حالته است. در زیر شمای کلی ازقفل مورد نظر در این قسمت مشاهده می شود.



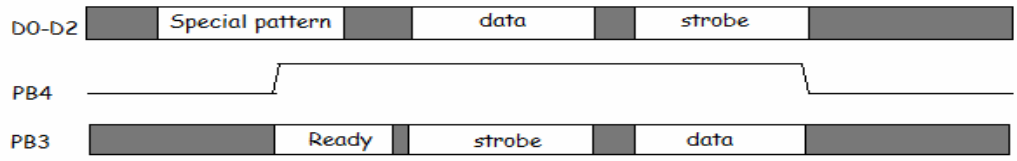
[14] Signal Monitoring

مطابق شکل، 3 بیت خروجی بین دستگاه جانبی و قفل سخت افزاری مشترک است. بیت PB3 برای خروجی میکروکنترلر در نظر گرفته شده است. همچنین برای کنترل بافرهای سه حالته برای جلوگیری از تداخل سیگنال ها از پایه PB4 استفاده شده است.

نحوه عملکرد این قفل به این صورت است که داده های D0-D2 همواره توسط میکروکنترلر پایش می شوند و در صورت مشاهده الگوی خاصی روی این 3 خط، میکروکنترلر برای کار آماده می شود و آمادگی خود را از طریق خط select اعلام می کند. همچنین برای جلوگیری از تداخل سیگنال ها، بافرهای 3حالته توسط میکروکنترلر به حالت امپدانس بالا می روند.

برای ارسال داده ها از PC به قفل از خطوط D0-D2 استفاده می شود و خط Select توسط میکروکنترلر برای انتقال سیگنال strobe استفاده می شود. پس از پردازش داده و تولید اعداد شبه تصادفی توسط میکروکنترلر، این اطلاعات از طریق خط select به صورت بیتی ارسال می شود و از خط D0 برای انتقال سیگنال Strobe از PC به میکروکنترلر استفاده می شود.

دیاگرام زمانی عملکرد قفل به صورت زیر خواهد بود.



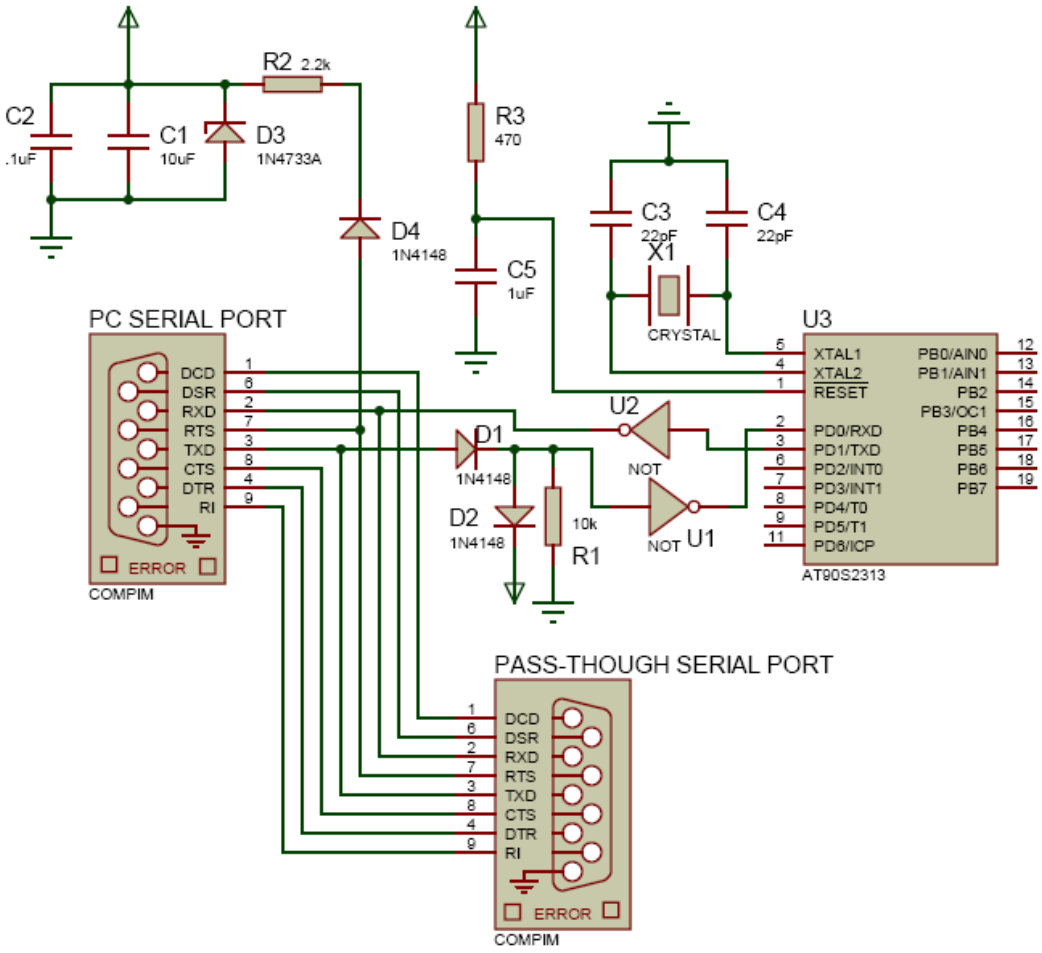
فصل چهارم: طراحی قفل سخت افزاری برای پورت سریال با استاندارد RS232

در این فصل قصد داریم روش قبلی که برای طراحی قفل سخت افزاری پورت موازی استفاده شد را به کار ببریم و با ارائه نرم افزار و سخت افزار جدید یک قفل سخت افزاری برای پورت سریال با استاندارد RS232 طراحی کنیم. بنابراین روش مورد استفاده برای تولید اطلاعات تصدیق قفل همان استفاده از LFSR است. قفل به گونه ای طراحی می شود که تغذیه مورد نیاز خود را از پورت دریافت کند و از آنجا که کامپیوترهای امروزی دارای یک پورت سریال هستند از مفهوم pass-through serial port استفاده می شود تا پورت سریال بین قفل سخت افزاری و سایر دستگاه های جانبی به اشتراک گذاشته شود.

برای رسیدن به اهداف فوق ملاحظات زیر در نظر گرفته شده است:

1- انتخاب میکرو کنترلر مناسب با کوچکترین ابعاد ممکن و دارای ورودی کریستال
2- پیکر بندی میکروکنترلر برای ارسال غیر همزمان
3- استفاده از کریستال خارجی برای دستیابی به دقت لازم برای ارسال غیرهمزمان زیرا استفاده از نوسان ساز داخلی میکروکنترلرهای AVR به دلیل نوسان تغذیه موجب بروز خطا خواهد شد
4- استفاده از سیگنال خروجی مناسب برای تغذیه قفل سخت افزاری
5- تبدیل سطوح ولتاژ از استاندارد RS232 به TTL و بالعکس

بر اساس این ملاحظات المان ها و شماتیک مدار مورد استفاده بدست آمده است که به همراه شماره آن ها در زیر مشاهده می شود. فرکانس کریستال 2,4576 مگاهرتز انتخاب شده است. این فرکانس برای انتقال اطلاعات با Baud rate برابر انتخاب شده است.



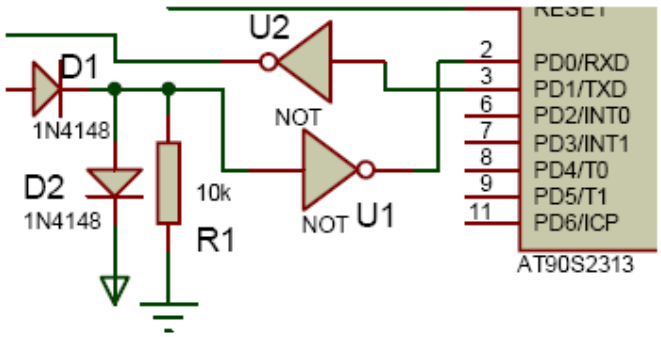
تمام خروجی های استاندارد RS232 در برابر اتصال کوتاه محافظت شده اند و می توانند جریانی بین 10 تا 20 میلی آمپر را sink یا source کنند. با این قابلیت می توان تغذیه مدارهای نسبتا کوچک از جمله قفل سخت افزاری مورد نظر را مستقیما از خروجی های واسط سریال تامین کرد. در این مدار از سیگنال خروجی RTS برای تغذیه استفاده شده است. این خروجی از طریق دیود محافظ و مقاومت محدود کننده جریان به یک دیود زنر 5,1 ولتی وصل شده است. همچنین از طریق دو خازن به عنوان فیلتر پایین گذر، تثبیت شده است.

مقاومت ورودی هر یک از ورودی ها حدود 10 کیلو اهم است و ولتاژ بالاتر از 1,25 ولت به عنوان حالت بالا و ولتاژ کمتر از 1 ولت نیز به عنوان حالت پایین شناخته می شود. ولتاژ های بین این دو ولتاژ شناخته نمی شوند و تغییر حالتی ایجاد نمی کنند. به طور معمول واسط سریال را توسط سیگنال دو قطبی با سطوح ولتاژ 12+ و 12- ولت راه اندازی می کنند اما با توجه به سطوح ولتاژ ذکر شده در بالا این امکان وجود دارد که این ورودی ها با سطوح ولتاژ TTL (5ولت/صفر ولت) نیز راه اندازی شوند[15]. بنابراین در اینجا به دلیل محدودیت در

[15] PC Interfaces، کانون نشر علوم

ابعاد مدار و تولید ولتاژ دو قطبی، از این امکان استفاده کرده و خروجی سریال میکروکنترلر را مستقیما به ورودی RS232 متصل می کنیم.

برای تبدیل ولتاژ خروجی RS232 به TTL و اعمال آن به ورودی میکرو از مدار شکل زیر استفاده شده است. وقتی که ولتاژ RS232 بالاتر از 5 ولت شود هر 2 دیود روشن شده و ولتاژ 5 ولت به ورودی میکرو اعمال می شود. همچنین وقتی ولتاژ RS232 از صفر کمتر شود هر 2 دیود خاموش شده و ورودی میکرو از طریق مقاومت pull-down به حالت پایین می رود. البته این روش دارای این عیب است که از سیگنال دریافتی یه صورت invert شده است بنابراین کلیه بیت ها از جمله start bit , stop bit و ... به صورت معکوس روی پایه ورودی میکرو قرار می گیرند. این عیب موجب می شود که نتوان از واحد UART داخلی میکروهایی که دارای چنین امکانی هستند استفاده کرد. برای رفع این مشکل می توان از یک گیت NOT استفاده کرد.



با توجه به اینکه پورت بین قفل و سایر دستگاه های جانبی به اشتراک گذاشته شده است، باید به نحوی میکرو کنترلر از در دست گرفتن پورت توسط نرم افزار محافظت شده آگاه شود. برای این منظور نرم افزار محافظت شده باید بعد از در درست گرفتن کنترل پورت سیگنال RTS را فعال کند تا میکرو کنترلر روشن و ریست شود بعد از آن با ارسال یک رشته از قبل تعیین شده حضور خود را اعلام کند. در این صورت میکرو نیز اعلام آمادگی کرده و نقل و انتقال داده انجام می شود. با این روش در صورتی که کنترل پورت سریال در دست سایر نرم افزار ها باشد، حتی با فعال بودن سیگنال RTS و روشن بودن میکروکنترلر، میکروکنترلر داده ای ارسال نخواهد کرد و خللی در کار سایر دستگاه های جانبی ایجاد نمی شود.

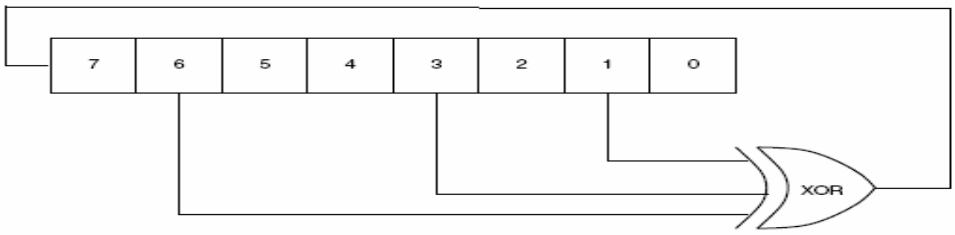
برای برنامه ربزی میکروکنترلر از زبان C استفاده شده است. این برنامه به همراه توضیحات در ادامه آورده شده است.

برای شبیه سازی عملکرد قفل نیز از نرم افزار Proteus 7 Professional استفاده شده است. در این برنامه یک ترمینال مجازی[16] وجود دارد که از این ترمینال برای شبیه سازی نقش کامپیوتر و نمایش فرآیند انتقال داده استفاده شده است.

[16] Virtual Terminal

نرم افزارمیکروکنترلر

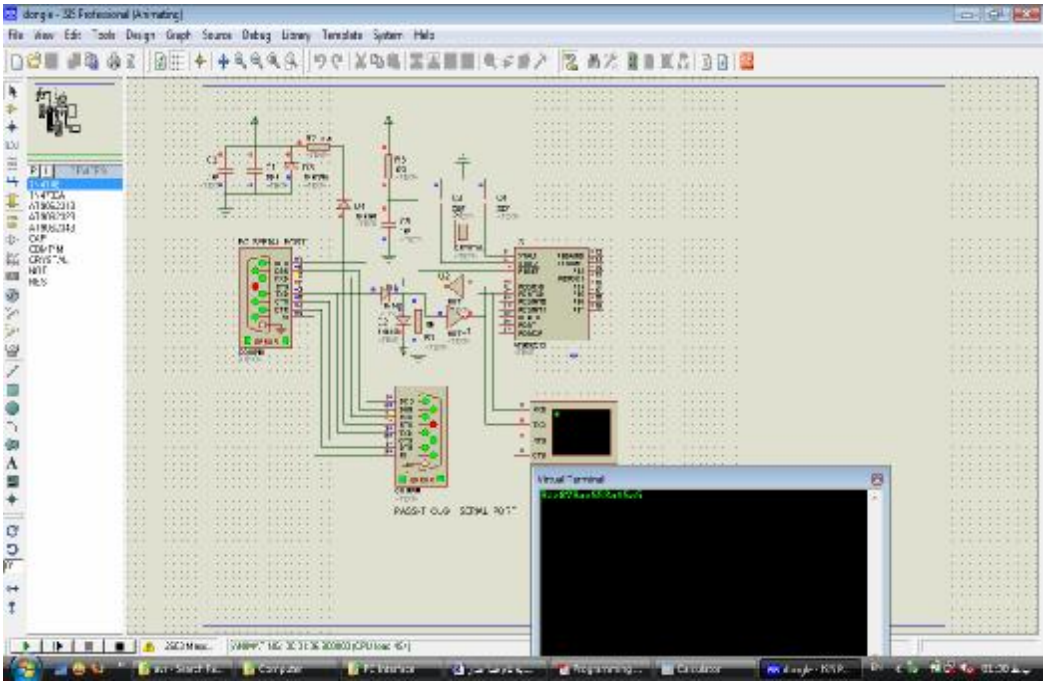
برنامه به گونه ای نوشته شده است که به محض مشاهده یک رشته خاص و از پیش تعیین شده روی خط داده با ارسال یک رشته خاص اعلام آمادگی می کند. پس از آن میکرو منتظر ارسال یک عدد اولیه از طرف کامپیوتر می شود. بعد از دریافت این عدد سه رقمی، میکروکنترلر شرع به ساخت و ارسال چندین عدد شبه تصادفی می کند. تعداد این اعداد و چندین پارامتر دیگر کاملا قابل تنظیم است. در برنامه یک LFSR به صورت زیر طراحی شده که تابع آن به صورت ساده شده در برنامه دیده می شود.



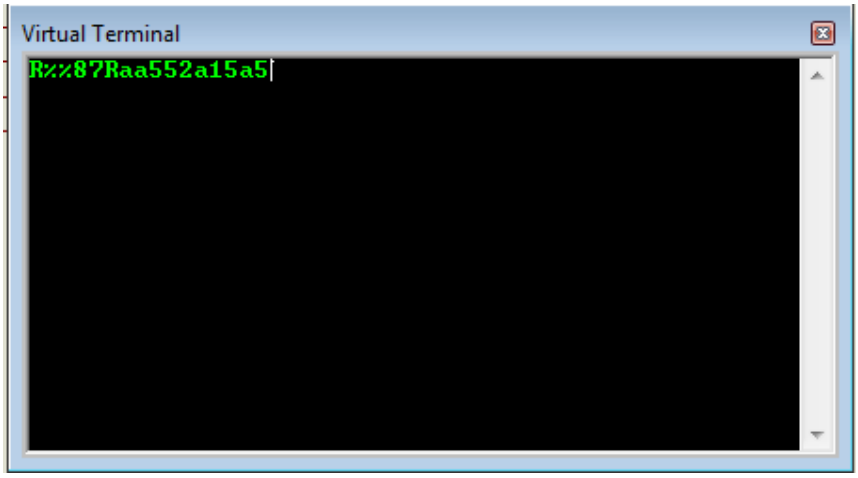

```
#include<90s2313.h>
#include<stdio.h>
#include<string.h>
#include<delay.h>
#define MAX_NUM_SEND 0x5;
void main()
{
int i;
char num;
unsigned char channel;
char str1[]="R%%87";// string to identify the
software on serial port
char in[5];
unsigned char DATA;
PORTD=0x00;
DDRD=0x00;
USR=0x0 ;
UCR=0b00011011;
UBRR=15;
while(1){
gets(in,5);
if(!strcmp(str1,in))putchar('R'); // if the software
is on port then 'R' is ready
signal from micro
getchar(num);
```

```
i=0;
LFSR(num,MAX_NUM_SEND) // calcute and send some
random number
}
int LFSR( char r,int n){
while(n>0){ //LFSR with 3 feedback line
if( ((num&&4) + (num&&16) + (num&&64))==2) { r=r>>1;
r=r|128;
}
else{ r=r>>1;
r=r&127;
}
printf("%c",r);
n=n-1;
}
}
}
```

نتایج شبیه سازی در زیر آمده است. برای نمونه عدد AAhex به عنوان عدد اولیه برای تولید اعداد تصادفی به میکرو فرستاده شده است. این کار توسط ترمینال مجازی صورت گرفته است. پس از آن میکرو اعداد 55، 2a ، 15 ، a و 5 را ارسال کرده است. این مراحل پس از انجام عملیات دست دهی انجام شده است.



در زیر جزئیات پنجره ترمینال مجازی قابل مشاهده است.



فصل پنجم: طراحی قفل سخت افزاری برای پورت USB

در این فاز نیازمند طراحی قفلی هستیم که توانایی ارتباط با پورت USB را دارا باشد. برای این منظور چندین راه وجود دارد. هر کدام از این راه ها دارای مزایا و معـایبی بـرای اسـتفاده در این پروژه هستند. درانتخاب یکی از این روش ها، به سادگی ساخت، قیمت ارزان، آسانی ارتقـا پروژه های قبلی برای استفاده در پورت USB و ... توجه شده است.

وسیله جانبی USB قاعدتا باید یک پورت USB و مداری برای ارتباط با میزبان[17] داشته باشد. فرستنده-گیرنده USB واسط سخت افزاری با باس ایجاد می کند. مدارهایی که با فرستنده-گیرنده ارتباط برقرار می کنند دارای نام عمومی، موتور واسـط سـریال (SIE) هسـتند. SIE فرستادن و دریافت داده های ترنزکشن[18] را به عهده دارد. این موتور داده هـای رسـیده را ترجمه می کند و فقط داده هایی را که برایش در دسترس قرار گرفته اند می فرستد و داده هایی را که رسیده اند نیز ذخیره می نماید. یک SIE عمومی باید همه کارهای زیـر را انجـام دهد[19]:

- تشخیص ورود یک پاکت[20]
- فرستادن پاکت ها
- تشخیص سگنال های شروع پاکت، انتهای پاکت، Reset و بازگشت
- رمزگذاری کردن و از رمز خارج کردن داده ها به قالب بندی ای که باس نیـاز دارد (NRZI به همراه bit stuffing)
- بررسی و تولید مقادیر CRC
- تشخیص و ایجاد شماره مشخصه های پاکت[21]
- تبدیل بین داده های سریال USB و داده های موازی رجیسترها و حافظه

انتخاب یک تراشه مناسب به کارایی، قیمت، دسترسی و راحتی ارتقا آن مربوط می شود.

در این پروژه ارتباط از طریق پورت USB با تراشه FT232 انجام می شـود. تراشـه FT232 یک تراشه USB USART جهت ارتباط به پورت USB است که توسط شـرکت FTD chip طراحی شده است . البته این شرکت تراشه دیگری بنام FT245 به بازار عرضه کرده که یـک واسط USB به پارالل میباشد.

از خصوصیات FT232 موارد زیر قابل ذکر است[22] :

- دارای رابط USART با سرعت و کیفیت بالا و عملکرد خودکار

---

[17] Host

[18] Transaction

[19] ، جان اکسلسون، کانون نشر علومUSB اصول و راهنمای استفاده از پورت

[20] Packet

[21] PID

[22] ، انتشارات نصAVR مرجع کامل میکروکنترلر های

- نرخ انتقال با لینک RS232 از 300 تا 1000 کیلو بیت بر ثانیه
- نرخ انتقال با لینک RS485 از 3000 کیلو بیت بر ثانیه
- 384 بایت بافر دریافت و 128 بایت بافر ارسال برای اطلاعات با حجم بالا
- انتقال خودکار بافر کنترل برای RS485
- مجهز به رگولاتور داخلی 3.3V
- پشتیبانی از روش ارسال توده ای و همزمان
- استفاده از کریستال خارجی 6MHz و افـزایش فرکـانس داخلـی تـا 48MHz بوسـیله ضرب کننده داخلی
- مجهز به خروجی نمایشگرهای ارسال و دریافت
- قابلیت تغییر شـماره مشخصـه فروشـنده (idvendor) و شـماره مشخصـه محصـول (idProduct) به دلخواه کاربر
- سازگار با نسخه های USB1.1 , USB2.0
- نیازمند یه یک حافظه EEPROM خارجی ، جهت ذخیره نمودن اطلاعـات مربـوط بـه idvendor,idProduct توصیفگرهای رشته محصول و شماره سریال

ارتباط بین میکرو و این تراشه از طریق پایه های RXD,TXD و RESET انجـام مـی پـذیرد. بنابر این برنامه میکرو باید در مد سریال نوشته شود. این ویژگی باعث می شود که ارتقا قفل سخت افزاری مربوط به پورت سریال بدون نیاز به دسـتکاری در نـرم افـزار میکـرو کنترلـر باشد. تغییرات سخت افزاری مورد نیاز در شماتیک مدار مـورد اسـتفاده قابـل مشـاهده مـی باشد.

تغذیه مدار از نوع USB Powered بوده و لذا نیاز به تغذیه خارجی نخواهیم داشت.

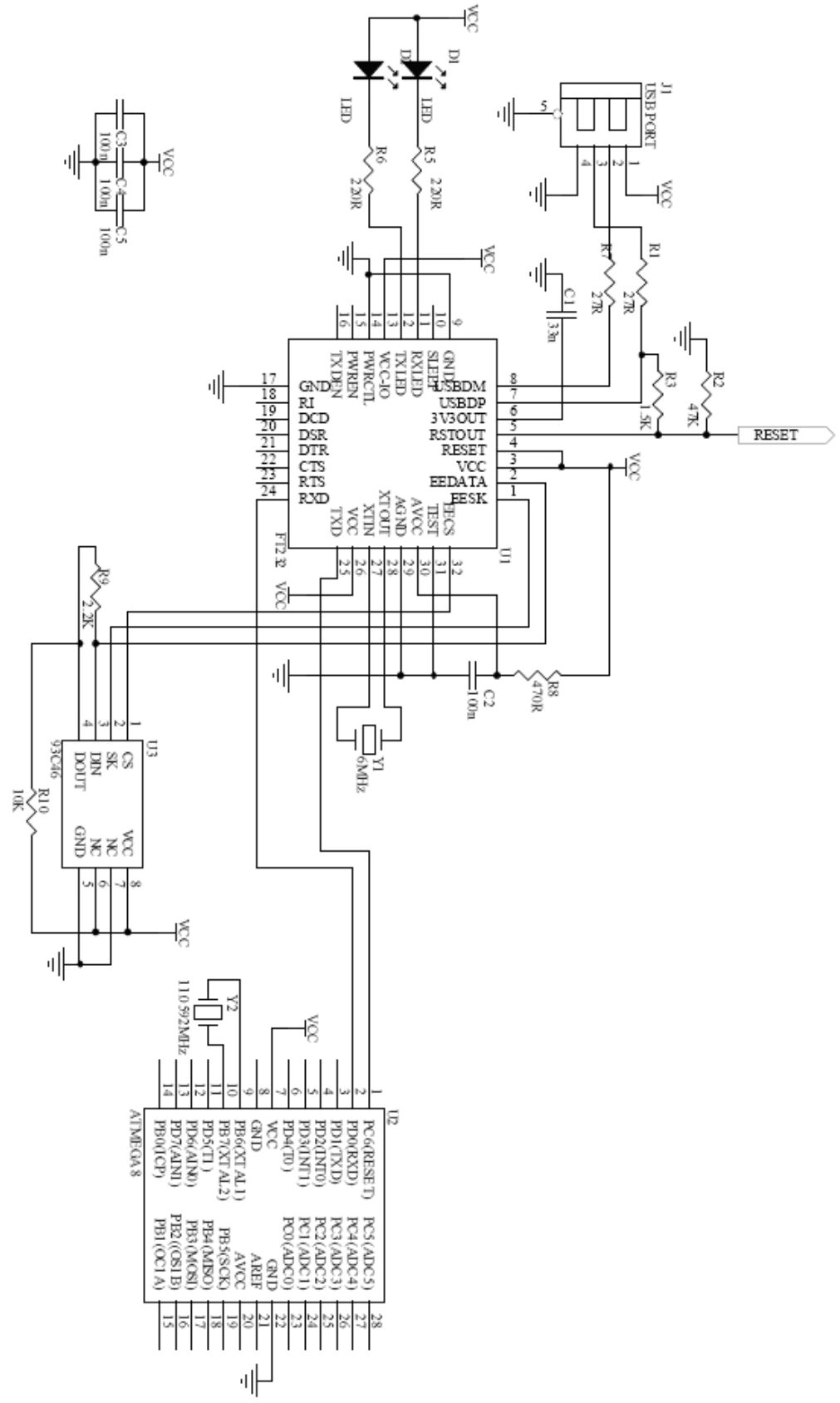
نرم افزارمیکروکنترلر

پس از ساخت مدار نوبت اتصال ان به کامپیوتر میباشد ودر صورتی که برای اولین بار آنرا به کامپیوتر متصل میکنید سیستم عامل درایور را پیدا نمی کند. بنابراین با گشوده شـدن پنجـره Add New Hardware این امکان به کاربر داده میشود تا خـود درایـور را بـه سیسـتم عامـل معرفی کند.

شرکت FTDI درایورهای مختلفی برای تراشه FT232 ارائه کرده است که کاربر با توجه بـه نیاز خود می تواند از آنها اسبفاده کند. در این پروژه ما از درایور ComPort استفاده میکنیم. درایور ذکر شده دستگاه را بعنوان یک درگاه سریال جدید به سیستم معرفـی میکنـد و بطـور مثال اگر سیستم شما دارای دو درگاه سریال با نامهای COM1 , COM2 باشد با این درایـور دستگاه بعنوان COM3 به سیستم معرفی میشود. بنابراین کار با پورت USB در نرم افزارهای

ویندوز به سادگی کار با درگاه RS232 خواهد بود. این ویژگی نیز باعث می شود که ارتقا قفل سخت افزاری مربوط به پورت سریال بدون نیاز به دستکاری در نرم افزار محافظت شده باشد.

همانطور که ذکر شد در این فاز از قفل سخت افزاری مربوط به پورت سریال استفاده خواهیم کرد و نیازی به ارتقا آن از لحاظ نرم افزاری وجود ندارد. اما سخت افزار آن به صورت زیر تغییر خواهد کرد.


J1
USB PORT
D1
LED
R6
220R
R5
220R
C3
100n
C4
100n
C5
100n
R7
27R
R1
27R
C11
33n
R3
1.5K
R2
47K
RESET
U1
FT232
GND
RI
DCD
DSR
DTR
CTS
RTS
RXD
TXDEN
PWREN
PWRCTL
VCC-IO
TXLED
RXLED
SLEEP
USBDM
USBDP
3V3OUT
RSTOUT
RESET
VCC
EEDATA
EESK
TEST
AVCC
AGND
XTOUT
XTIN
VCC
TXD
C2
100n
R8
470R
Y1
6Mhz
R9
2.2K
U3
93C46
CS
SK
DIN
DOUT
VCC
NC
NC
GND
R10
10K
Y2
11.0592Mhz
U2
ATMEGA8
PB6(XTAL1)
PB7(XTAL2)
PD5(T1)
PD6(AIN0)
PD7(AIN1)
PB0(ICP)
PB1(OC1A)
PB2((SS1B)
PB3(MOSI)
PB4(MISO)
PB5(SCK)
AVCC
AREF
GND
VCC
PD4(T0)
PD3(INT1)
PD2(INT0)
PD1(TXD)
PD0(RXD)
PC6(RESET)
PC5(ADC5)
PC4(ADC4)
PC3(ADC3)
PC2(ADC2)
PC1(ADC1)
PC0(ADC0)

فصل ششم: جمع بندی

در این نوشته ابتدا به بیان تعاریف و مسائل مقدماتی در مورد قفل های سخت افزاری پرداخته شد. در این پروژه 4 نوع قفل سخت افزاری مختلف طراحی و شرح داده شد که برای سه پورت مختلف سیستم های امروزی قابل استفاده اند. در اینجا قصد داریم به بیان ویژگی های مثبت و منفی هر یک بپردازیم و مقایسه ای بین آن ها انجام دهیم.

با نگاهی به سیستم های امروزی در می یابیم که پورت USB در این سیستم بیشترین تعداد را به خود اختصاص داده است. دلیل گسترش USB را می توان به طور کلی در موارد زیر خلاصه کرد:

- سادگی استفاده برای توسعه
- راه حل ارزان برای انتقال سریع داده
- حمایت از کار همزمان چندین دستگاه جانبی
- پنهان شدن جزئیات الکتریکی از دید کاربر
- ایجاد قابلیت plug & play
- استفاده از کابل و کانکتور یکسان
- و...

با اندکی تامل در موارد بالا درمیابیم که پورت USB می تواند به طور همزمان تسهیلاتی را برای کاربر و طراح دستگاه های جانبی فراهم آورد. طراحان و استفاده کنندگان از قفل های سخت افزاری نیز از این تسهیلات بی نصیب نیستند و بازار قفل های سخت افزاری نیز این مسئله را تایید می کند. از جمله تسهیلاتی که پورت USB برا ی طراحان قفل سخت افزاری فراهم می کند، می توان به موارد زیر اشاره کرد:

- سادگی ارتقا قفل های قبلی متصل شونده به پورت های سریال و موازی برای اتصال به پورت USB
- کاهش پیچیدگی های مداری برای ایجاد امکان استفاده همزمان دستگاه های جانبی دیگر از یک پورت علاوه بر قفل سخت افزاری
- یکسان بودن کانکتور های پورت های USB
- و ...

بزرگترین عیب قفل های سخت افزاری مبتنی بر پورت USB قیمت گران تر آن ها نسبت به سایر قفل ها است.

با نگاهی به شماتیک مدار های قفل های ارائه شده این موضوع به راحتی آشکار می شود که طراحی مدار برای پورت USB ساده تر از سایر پورت ها خواهذ بود. همچنین در صورت

استفاده از USB Controller های ارائه شده توسط شرکت های مختلف، طراحی نـرم افـزار های مربوط به میکروکنترلر و PC نیز تا حدودی ساده خواهد شد.

در آخر می توان نتیجه گرفت که با توجه بـه شـرایط کنـونی اسـتفاده از پـورت USB بـرای طراحی قفل سخت افزاری بهترین گزینه بـرای خوهـد بـود. همچنـین در آینـده ای نزدیـک طراحان قدیمی برای حفظ امکان رقابت در بازار قفل هـای سـخت افـزاری بـه ناچـار مجبـور خواهند بود که قفل های قبلی خود را برای استفاده در پورت USB ارتقا دهند یا طـرح هـای جدیدی مبتنی بر این پورت ارائه دهند.

مراجع

۱. مرجع کامل میکروکنترلرهای AVR، انتشارات نص
۲. اصول و راهنمای استفاده از پورت USB، جان اکسلسون، کانون نشر علوم
۳. PC Interfaces، کانون نشر علوم

5. Programming and customizing the avr microcontrollers by DHANAJAY V.GADRE
6. www.atmel.com
7. www.mavidesigne.com/glossary.html
8. www.theidm.com/index.cfm
9. www.wikipedia.com
10. www.wisegreek.com/what_is_a_dongle.html
11. www.iritn.com