علیرضا اسلامی خواه

99521064

تکلیف سوم ریزپردازنده

بخش تئوری

سوال 7)

یک تایمر اغلب می تواند یک پین خروجی را هر بار که سرریز می کند تغییر دهد. اگر تایمر با همان فرکانس ریزپردازنده کار کند (یعنی بدون پیش مقیاس‌کننده)، تا حداکثر مقدار خود شمارش می‌کند و سپس سرریز می‌کند و هر بار پین را تغییر می‌دهد.

برای یک تایمر 8 بیتی، شمارنده از 0 به 255 (xFF0) می‌رود و سپس روی 0 می‌چرخد و در هنگام سرریز، پین خروجی را تغییر می‌دهد. بنابراین، در هر چرخه شمارش کامل، 256 بار پین خروجی را تغییر می دهد. از آنجایی که فرکانس موج نصف فرکانس جابجایی است (یک موج مربعی از یک زیاد و یک کم برای هر چرخه تشکیل شده است)، بالاترین فرکانس موج مربعی خواهد بود:

```
Frequency = (microcontroller frequency) / (2 * counter max value)
Frequency = 8 MHz / (2 * 256)
Frequency = 8,000,000 Hz / 512
Frequency = 15,625 Hz
```

بنابراین، بالاترین فرکانس موج مربعی در یک تایمر 8 بیتی با فرکانس میکروکنترلر 8 مگاهرتز، 15625 هرتز خواهد بود. با این حال، اگر میکروکنترلر دارای یک تایمر 16 بیتی باشد، می توانید فرکانس بالاتری دریافت کنید زیرا مقدار حداکثر شمارنده بزرگتر است.

سوال 1)

با توجه به اینکه حافظه برنامه در میکروکنترلرهای AVR دارای آدرس کلمه است و با دانستن اینکه .ORG 0x100 شروع OUR_DATA را تعریف می کند، آدرس در قالب آدرس بایتی x1000 خواهد بود. با این حال، از آنجایی که دستورالعمل AVR LPM هنگام واکشی از حافظه برنامه از آدرس های کلمه استفاده می کند و دستورالعمل .EQU در کد استفاده می شود، تغییر آدرس یک بیت به سمت چپ آدرس بایت را به آدرس کلمه تبدیل می کند.

بنابراین، آدرس بایت مستقیم x1000 به آدرس کلمه x1000 در فضای آدرس‌دهی کلمه تبدیل می‌شود (زیرا جابه‌جایی یک بیت به چپ مقدار را دو برابر می‌کند). در شرایط بایت آدرس دهی، آدرس پس از تغییر x2000 خواهد بود. این معمولاً یک خطا است زیرا دستورالعمل .ORG 0x100 نشان می دهد که داده ها از آدرس بایت x1000 شروع می شوند نه x2000. به نظر می رسد که در فرض آدرس دهی حافظه خطایی وجود دارد، اما اطلاعات را همانطور که داده شد ادامه می دهیم.

محاسبه بایت های کم و زیاد x200:

LOW(0x200) = 0x00 (زیرا x200 0b 0000 0010 0000 00000 در باینری است)

HIGH(0x200) = 0x02 (زیرا x200 0b 0000 0010 0000 00000 به صورت باینری است)

بنابراین:

R30 = 0x00

R31 = 0x02

با داشتن R30 و R31 این مقادیر، رجیستر Z 0x0200 خواهد بود.

دستورالعمل LPM R20، Z محتوای حافظه برنامه را در آدرسی که توسط Z اشاره شده در R20 بارگذاری می کند. بدون اطلاع از کل محتوای حافظه برنامه، ما فقط می دانیم که سعی می کند از آدرس x2000 بایت (یا آدرس x1000 کلمه در صورت آدرس کلمه) بخواند. اگر فرض کنیم OUR_DATA شروع حافظه برنامه را نشان می‌دهد، یک CPU

AVR نمی‌تواند بارگیری را از حافظه برنامه با استفاده از LPM با آدرسی که به دستورالعمل‌های .DB اشاره دارد، اجرا کند، زیرا برای داده‌های ثابت در نظر گرفته شده است. علاوه بر این، LPM از حافظه فلش می خواند نه از SRAM، و معمولاً .DB برای داده ها در SRAM استفاده می شود که کار را بیشتر پیچیده می کند.

بر اساس کدی که داریم، با استفاده از دستورالعمل LPM از حافظه فلش در آدرس کلمه ای که از مقدار جابجا شده است خوانده می شود، اما از آنجایی که محتوایی برای آنچه در x2000 (آدرس کلمه) در حافظه است (فلش) نداریم. ، ما نمی توانیم تعیین کنیم که R20 به چه مقداری می رسد. در یک CPU واقعی AVR، R20 حاوی هر چیزی است که در آن آدرس در Flash وجود دارد.

بنابراین با اطلاعات داده شده:

R20 = تعریف نشده/ناشناخته (بستگی به محتویات حافظه فلش در آن آدرس دارد)

R30 = 0x00

R31 = 0x02

DATA_ADDR = 0x200 (پس از تصحیح شیفت، با فرض آدرس کلمه برای حافظه برنامه که برای LPM معمولی است)

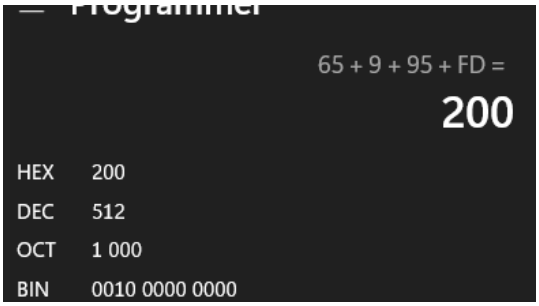
سوال 2)

الف) ابتدا اعداد را با هم جمع میکنیم :

```
0x65 + 0x09 + 0x95 = 0x103
```

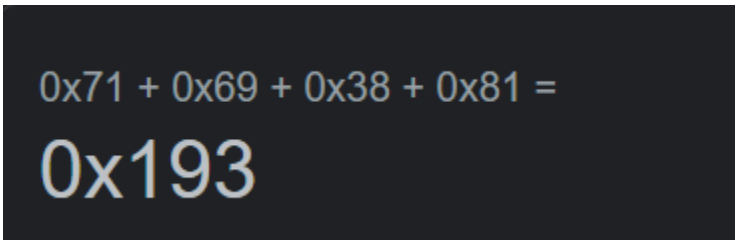
پس در اینجا 0x03 را برای 2 s complement ’ انتخاب میکنیم. که میشود : 0xFD

و در آخر هم جمع اعداد 200 میشود :

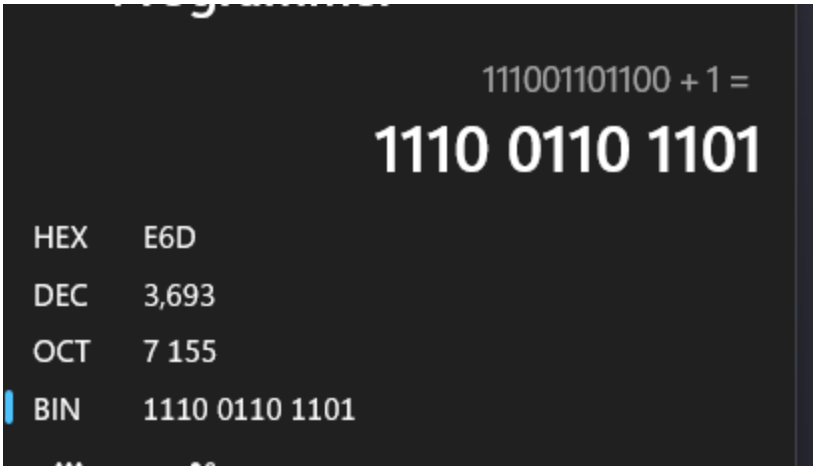


که در آخر 00 در آخر عدد هگز نشان میدهد دیتا corrupt نشده است.

ب) مراحل مانند بالا :



2s complement :



و در آخر :

که دو بیت سمت راست نشان میدهد دیتایی corrupt نشده است.