**"به نام یزدان پاک"**

**گزارش آزمایش اول اسمبلی**

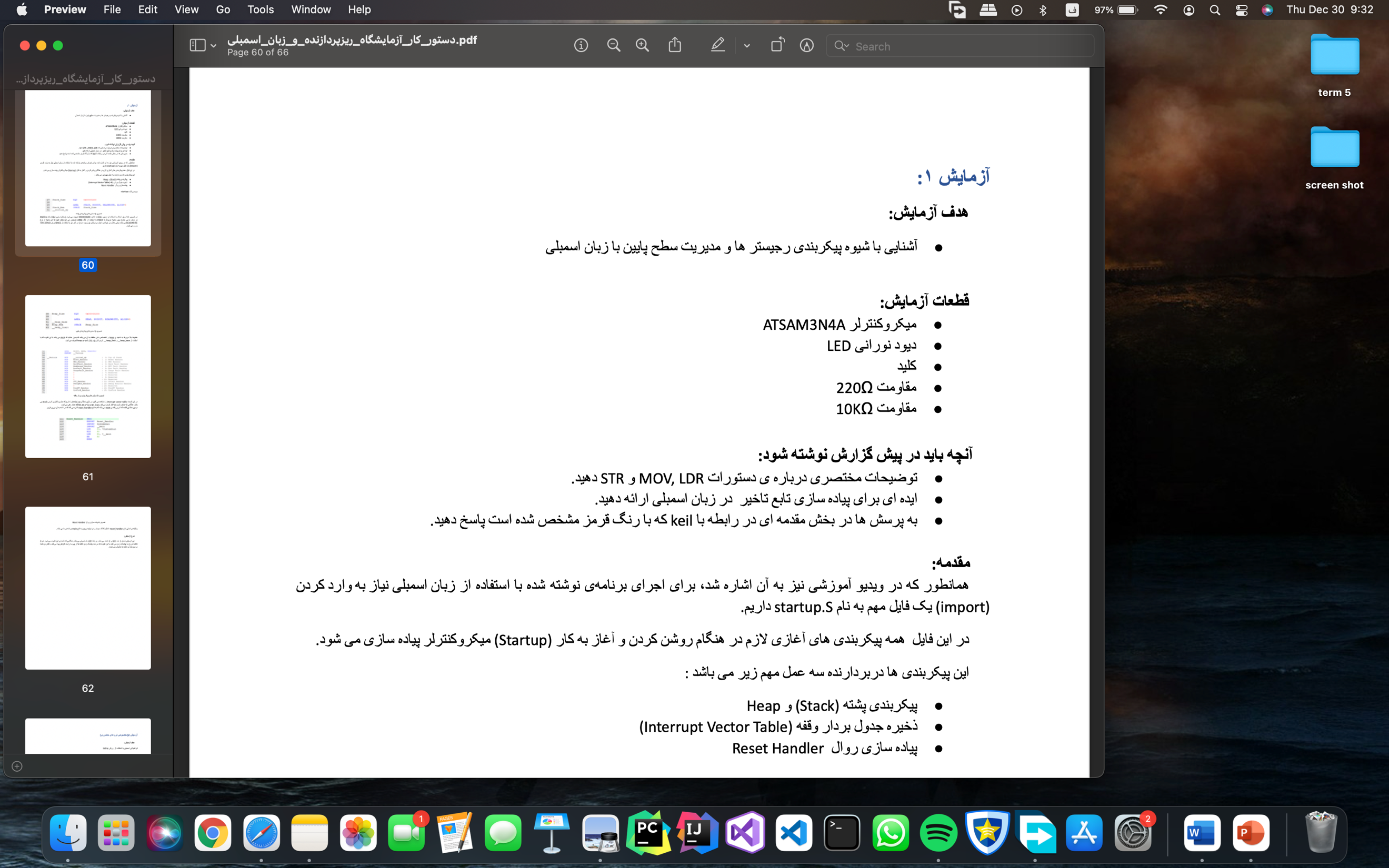
**اعضای گروه:**

**آرتا اسدی حقی 9731006**

**کیانا آقاکثیری 9831006**

**سارا تاجرنیا 9831016**

**تاریخ تحویل گزارش :9/10/1400**



قسمت اول:

**LDR**

**دستورالعمل** رجیستر **LDR** ، دیتای مستقر در یک آدرس را درون رجیستر مقصد بارگذاری میکند. پرانتز های اطراف R1 حاکی از این است که رجیستر یک آدرس دارد. ما با استفاده از پرانتز ها دیتا را به جای خود آن آدرس در آدرس R0 قرار میدهیم. همچنین میتوانیم از این علامت گذاری برای قرار دادن دیتای offset از یک آدرس معین استفاده کنیم، همانطور که در **دستورالعمل** خط دوم نشان داده شده است. R0 شامل دیتای دو کلمه ای دور از هر چیزی است که دارای آدرس R0 است.

LDR R0, [R1]

LDR R0, [R1, #8]

**MOV**

عمل حرکت (MOV) دقیقا همان کاری که به نظر می آید را انجام میدهد. یعنی دیتا را از جایی به جای دیگر منتقل میکند. در دستور زیر، R1 به R0 کپی میشود. در خط دوم مقدار فوری 8 در R0 قرار میگیرد.

MOV R0, R1

MOV R0, #8

**STR**

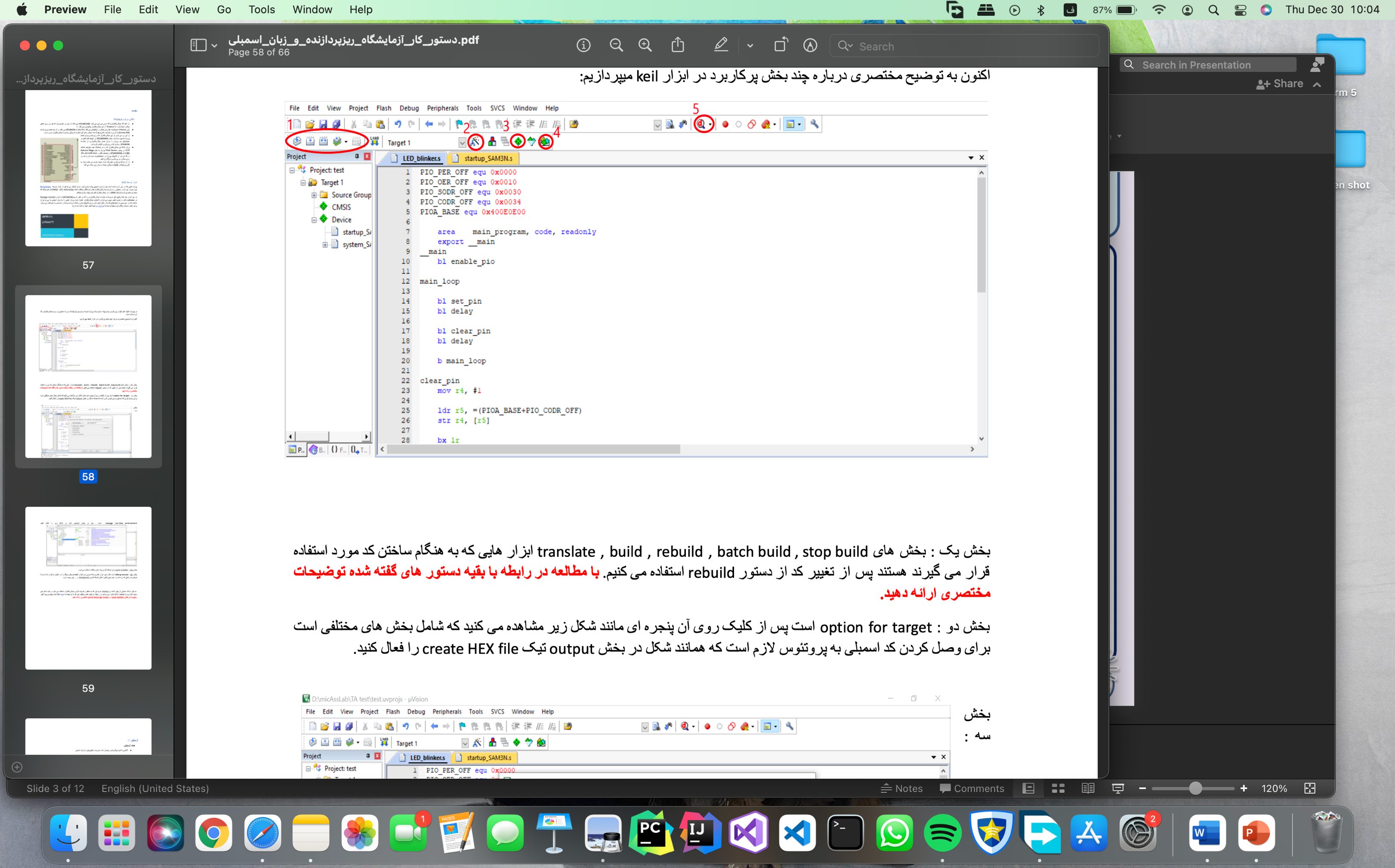
ذخیره سازی یک عملیات مکمل را برای بارگیری انجام میدهد. ذخیره سازی، محتویات رجیستر را در محل حافظه قرار میدهد. کد زیر دیتای درون R1 را در آدرس R0 ذخیره میکند. پرانتز ها باز هم نشان دهنده ی این هستند که R0 یک آدرس دارد و ما میخواهیم دیتای آن آدرس را تغییر دهیم.

STR R1, [R0]

قسمت دوم:

برای پیاده سازی تاخیر در زبان اسمبلی برای مثال میتوان یک loop برای مقداری مشخص مثلا ۱۰۰۰ در نظر گرفت که تا زمانی که EQU باشد در این loop همچنان میچرخد و این موضوع سبب ایجاد dealy میشود.

حال به مقدار لازم میتوان رجیستری که EQU آن هر دفعه چک میشود را مشخص کرد.

قسمت سوم:

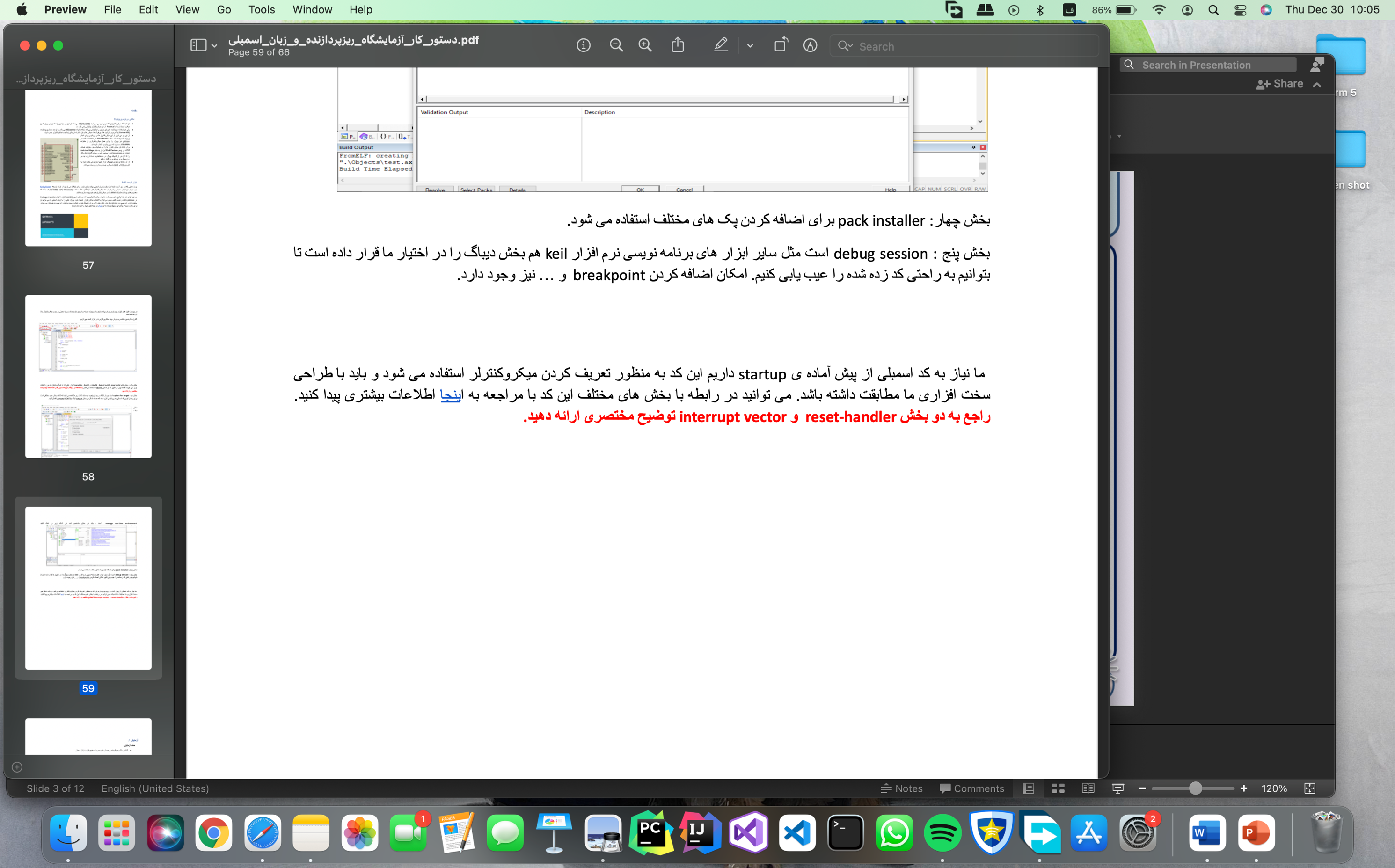
build: پروژه نوشته شده در برنامه را اجرا میکند.

Rebuild: برنامه را برای دفعات بعدی از ابتدا دوباره built میکند.

Batch build: برنامه را از نقطه مشخصی اجرا میکند(معمولا برای debug کردن اجرا میشود)

Stop build: برنامه را در میانه کار متوقف میکند.

Translate: برنامه اسمبلی را به سطحی تبدیل میکند که ماشین آن را به طور کامل متوجه شود.



Reset-handler: برای زمانی که برنامه را reset میکنیم برنامه را load میکند و اجزای مختلف برنامه را initialize میکند. مراحل زیر طی میشوند:

1) After reset, PC is loaded with the address 0x00000000

2) Then the processor fetches the value at 0x00000000 in to the MSP. i.e Main Stack Pointer.

3) Next the processor reads the address of the reset handler from the location 0x00000004 in to the Program Counter.

4) Then the processor jumps to your reset handler and start executing the first instruction which you wrote there .

5) After the required initialization, you can then call your main() from Reset Handler ,that’s how the control comes to your main() function in your Application .

Interrupt-handler: به بررسی interrupt ها میپردازد برای مثال وقفه I/O یا از جمله کارهای زیر:

Sigle-channel DMA transfar

Dual-channel DMA transfar

Interrupt prioritization

Context switch