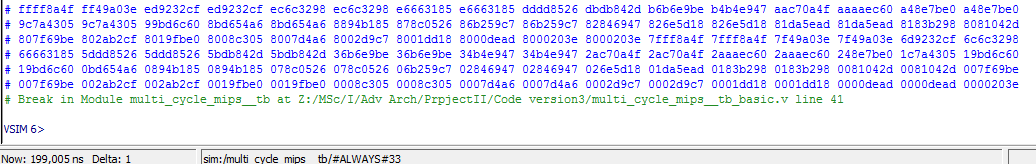


بخش دوم: اجرای کامل برنامه

در این بخش خروجی برنامه به شکل زیر است:



اجرای کامل برنامه حدود 199 نانوثانیه به طول می‌انجامد که حدود 1.96 برابر بهبود در سرعت اجرای پایپ‌لاین عادی است.

قابلیت های بعدی (شامل قابلیت اضافه شدن cache‌ و branch prediction)

**در مورد قابلیت اضافه شدن cache، طراحی اولیه این برنامه همراه کش بود اما به دلیل ساده تر شدن کار این کش حذف شد. برای این کار صرفا نیاز به استفاده از icache داریم تا دستورات یک کلاک زودتر در اختیار ما قرار بگیرند. این قابلیت وجود دارد تا زیر کنترل یونت این cache اضافه گردد.**

**قابلیت branch prediction در این برنامه وجود ندارد اما جای آن بسیار خالی است و قطعا قابلیت اضافه شدن را دارد. اگر از icache استفاده کنیم این واحد می‌تواند کنار کنرل یونیت مشغول به کار شود.**

اشتباهات و نکات آموزنده

در مورد نکات آموزنده، مهم ترین نکته این است که برای کارهای وسیع این چنینی حتما از دیاگرام بلوکی واضح و صریح استفاده شود تا از سردرگمی و فراموشی جلوگیری گردد. همچنین تا جای ممکن در برخی کارها که ارتباط بین دو واحد زیاد است، اگر این دو واحد درون یک بخش قرار دارند منطقی است که این دو در دو ماژول جداگانه نوشته نگردند، چرا که برای صحت دریافت و ارسال سیگنال‌های صحیح نیاز به استفاده از تاخیر می‌گردد که می‌تواند روند اجرای برنامه را مختل کند.

در مورد اشتباهات انجام شده به طور خیلی خلاصه، اشتباهات زمانبر عبارت بودند از:

* با توجه به حساس بودن وریلاگ به بزرگی و کوچکی حروف، بسیار مختمل است که یک وایر که برای ارتباط بین دو ماژول است را سهوا تعریف نکرده باشیم و یا اسم آن را اشتباهی با یک حرف بزرگ و کوچک جابجا نوشته باشیم، در این حالت متاسفانه کامپایلر، این متغیر را یک وایر یک بعدی در نظر گرفته، که در حالتی که در ماژول کپی شده هم این متغیر یک بعدی باشد بدون هیچ خطا و هشداری برنامه شبیه سازی شده و مشغول به کار می‌شود.
* بهتر است برای هر چیزی بدترین حالت در نظر گرفته شود، مثلا برای Reg\_stat اشتباها در ابتدا برای هر رجیستر 8 خانه (مربوط به 4 دستورالعمل) در نظر گرفته شده بود که مشکلات زیادی را به وجود آورد. با توجه به اینکه 8 دستورالعمل حداکثر از یک رجیستر 24 بار استفاده می‌کنند و 48 خانه نیاز است ، لذا 64 خانه برای هر رجیستر اختصاص داده شد.
* در یک حلقه ای که تعداد خاصی انجام می‌شود، اگر تاخیر استفاده می‌شود خیلی مهم است که این تاخیر ها طوری تنظیم شوند که قبل از تحریک مجدد بلاک always این حلقه تمام شده باشد.
* استفاده از متغیرهای یکسان در بلاک های always مختلف ایجاد خطای ران تایم می‌کنند.