

آز سیستم های دیجیتال

دکتر اجلالی

میبا حیدری، عاطفه قندهاری، نیکا قادری
بهار ۱۴۰۳



آزمایش هفتم

UART

تاریخ گزارش: ۲۶ اردیبهشت ۱۴۰۳

۱. شرح آزمایش

در این آزمایش می خواهیم یک UART یا Transmitter Receiver Asynchronous Universal طراحی کنیم. در بخش فرستنده این UART هر بار ۱۰ بیت ارسال میشود که بیت اول برای شروع، بیت دوم برای توازن، ۷ بیت بعدی برای داده و نهایتاً بیت آخر برای خاتمه ارسال می شوند. در بخش گیرنده این UART پس از دریافت بیت شروع، ۸ بیت بعدی یعنی توازن و داده به صورت سریالی دریافت شده و هر یک در ثباتی مجزا ذخیره می شوند.

۲. ماژول ها

Sender

ورودی ها:

- rstN
- clk
- start: برای شروع ارسال برابر با ۱ می شود.
- data_in: داده ای که قرار است ارسال شود.

خروجی ها:

- s_out: خروجی سریالی فرستنده.
- Sent: با اتمام ارسال، برابر با ۱ می شود.

این ماژول، فرستنده ی UART است .

برای این ماژول، ۵ حالت مختلف داریم idle, start, parity, send, stop

چون مدار ترتیبی است، از block always استفاده میکنیم که به لبه بالارونده کالک و لبه پایین رونده ریست حساس است. در صورت شدن سیگنال rstN، فرستنده در حالت idle قرار می گیرد، اندیس داده برابر با ۰ می شود و خروجی ها ۰ می شوند. در غیر اینصورت با لبه بالارونده کالک، بر حسب حالتی که در آن قرار داریم عمل می کنیم:

Idle: اگر ورودی start برابر با ۱ شده باشد، وارد حالت بعدی یعنی start می شویم، ورودی in_data در رجیستر data قرار می گیرد و اندیس داده و خروجی sent برابر با صفر می شوند. در غیر اینصورت اتفاقی نمی افتد.

Start: بیت شروع که برابر با ۱ است در خروجی سریالی قرار میگیرد و به حالت parity میرویم.

Parity: مقدار parity که برابر با حاصل xor بیت های data است در خروجی سریالی قرار می گیرد و به حالت send می رویم

Send: بیتی که در اندیس index_data از رجیستر data قرار دارد، وارد خروجی سریالی میشود، اندیس داده یکی بیشتر می شود و اگر برابر با آخرین اندیس داده که ۶ است بشود، از حالت send خارج شده و به stop می رویم. در غیر اینصورت در همین حالت میمانیم.

Stop: بیت پایان که برابر با ۰ است به خروجی سریالی می رود، به حالت idle باز میگردیم و خروجی sent برابر با ۱ میشود، چون ارسال به پایان رسیده است.

اگر غیر از این ها بود وارد حالت idle می شویم.

Receiver

ورودی ها:

- rstN
- clk
- s_in: ورودی سریالی.

خروجی ها:

Received: در صورت پایان گرفتن ورودی ها، ۱ می شود.

parity_chack: در صورت برابر parity ارسالی با parity داده ی گرفته برابر بود، ۱ می شود.

• Data: داده ی گرفته شده.

این ماژول، گیرنده ی UART است.

برای این ماژول ۴ حالت مختلف داریم idle, parity, receive, stop :

چون مدار ترتیبی است، از block always استفاده میکنیم که به لبه بالرونده کالک و لبه پایین رونده ریست

حساس است. در صورت ۰ شدن سیگنال rstN، فرستنده در حالت idle قرار می گیرد، اندیس داده برابر با ۰ می شود

و خروجی ها ۰ می شوند.

در غیر اینصورت با لبه بالرونده کالک، بر حسب حالتی که در آن قرار داریم عمل می کنیم:

idle: اگر ورودی سریالی با بیت شروع یعنی ۱ برابر بود، وارد حالت parity می شویم و اندیس داده و

خروجی ها برابر با ۰ میشوند. در غیر اینصورت اتفاقی نمی افتد.

Parity: مقدار ورودی سریالی که parity ارسال شده است، در parity_expected قرار گرفته و به حالت

receive می رویم.

Receive: مقدار ورودی سریالی در اندیس index_data از رجیستر data قرار می گیرد، اندیس داده یکی

بیشتر می شود و اگر برابر با آخرین اندیس داده که ۶ است بشود، از حالت send خارج شده و به stop

میرویم. در غیر اینصورت در همین حالت میمانیم.

Stop: به حالت idle باز میگردیم و خروجی received برابر با ۱ میشود، چون گرفتن داده به پایان رسیده

است.

• اگر غیر از این ها بود وارد حالت idle می شویم.

Uart

ورودی ها:

- rstN
- clk
- s_in
- send
- send_data

خروجی ها:

- s_out
- sent
- received
- received_data

check_receive_parity•

Top

در این ماژول به بررسی کارکرد این دستگاه می پردازیم. ابتدا از ماژول Uart دو شیء می سازیم .برای تست اول، از فرستندهی اول برای گیرنده ی دوم پیام Hello را می فرستیم. برای تست دوم از فرستنده ی اول پیغام Bye را ارسال می کنیم و ن تایچ را در transcript و waveform مشاهده خواهیم کرد.

۴. waveform

حاصل شبیه سازی ماژول Top :

