[[CRC] Bài 4 - Bộ tính CRC song song từng phần](https://nguyenquanicd.blogspot.com/2019/09/crc-bai-4-bo-tinh-crc-song-song-tung.html)

Tác giả [Nguyễn Quân](https://www.blogger.com/profile/11739939602738380602) at 07:58  [CRC](https://nguyenquanicd.blogspot.com/search/label/CRC?&max-results=5), [Synchronous Design](https://nguyenquanicd.blogspot.com/search/label/Synchronous%20Design?&max-results=5), [Verilog](https://nguyenquanicd.blogspot.com/search/label/Verilog?&max-results=5)  [1 bình luận](https://nguyenquanicd.blogspot.com/2019/09/crc-bai-4-bo-tinh-crc-song-song-tung.html#comment-form)

*Bài viết này trình bày một cách thiết kế bộ tính CRC song song nhưng có chút khác biệt so với bài [bài 3](http://nguyenquanicd.blogspot.com/2019/08/crc-bai-3-bo-tinh-crc-song-song.html" \t "_blank). Để giảm số chu kỳ tính toán CRC nhưng không tạo ra mạch tổ hợp quá lớn, chuỗi dữ liệu ngõ vào sẽ được chia nhỏ thành từng đoạn để tính CRC. Cách tính này tạm gọi là tính CRC song song từng phần.*

**1) Nguyên lý tính CRC từng phần**

Việc tính CRC nối tiếp như [bài 1](http://nguyenquanicd.blogspot.com/2017/10/basic-knowledgecrc-bai-1-ly-thuyet-ve.html" \t "_blank) hoặc [bài 2](http://nguyenquanicd.blogspot.com/2019/08/crc-bai-2-logic-tinh-crc-noi-tiep-voi.html" \t "_blank) làm số chu kỳ tính toán cho một chuỗi dữ liệu lớn. Việc tính CRC song song toàn phần, toàn bộ chuỗi dữ liệu được đưa qua mạch tính CRC và cho kết quả CRC trong một lần tính, làm cho mạch tổ hợp tính CRC lớn ([bài 3](http://nguyenquanicd.blogspot.com/2019/08/crc-bai-3-bo-tinh-crc-song-song.html" \t "_blank)).

Một phương pháp giúp cân bằng giữa số chu kỳ tính CRC và độ lớn của mạch tổ hợp tính CRC là tính song song từng phần. Với cách tính này, chuỗi dữ liệu ngõ vào sẽ được chia nhỏ thành từng phần với một độ dài xác định. Mỗi phần này sẽ được đưa vào mạch tính CRC song song.  
Nguyên lý tính CRC song song từng phần được thực hiện như sau:

1. Chia chuỗi dữ liệu ngõ vào thành từng nhóm có số bit bằng với số bit của chuỗi CRC
2. Thực hiện XOR một nhóm bit của chuỗi dữ liệu đầu vào với giá trị chuỗi CRC hiện tại. Với lần XOR đầu tiên, giá trị chuỗi CRC hiện tại là giá trị khởi tạo ban đầu. Trong bài viết này, giá trị này bằng 0.
3. Kết quả phép XOR ở bước 2 được đưa đến mạch tính CRC song song. Kết quả của bước này sẽ được lưu lại để sử dụng cho lần tính tiếp theo. Lần tính tiếp theo là lần tính cho nhóm bit kế tiếp và lặp lại từ bước 2.

|  |
| --- |
|  |
| Hình 1: Minh họa nguyên lý tính CRC từng phần |

**2) Phân tích thiết kế**

Nguyên lý thiết kế như đã trình bày sẽ được cụ thể hóa trong hình minh họa sau đây với số bit dữ liệu đầu vào là 32, DWIDTH=32, và số bit CRC là 8, CRC\_WIDTH=8.

|  |
| --- |
|  |
| Hình 2: Sơ đồ nguyên lý mạch tính CRC từng phần với trường hợp DWIDTH=32 và CRC\_WIDTD=8 |

Trong sơ đồ trên, các bước tính toán CRC như sau:

* (1) Store input data: Chuỗi dữ liệu đầu vào được lưu vào một thanh ghi trước khi bắt đầu tính CRC, trong ví dụ này, đây là thanh ghi 32 bit. Nó là một thanh ghi dịch. Thanh ghi sẽ dịch *CRC\_WIDTH*bit sau mỗi chu kỳ xung clock. Trong ví dụ này, nó dịch 8 bit.
* (2) Create input: Mỗi nhóm 8 bit sẽ được XOR với giá trị CRC hiện tại lấy từ thanh ghi *crcSeq[CRC\_WIDTH-1:0]*. Sau đó ráp với một chuỗi giá trị khởi tạo bằng số bit chuỗi CRC, trong ví dụ này nó là 8 bit 0. Phần logic này là mạch tổ hợp.
* (3) Parallel calculation: Là một mạch tính CRC cho nhóm 8 bit dữ liệu và 8 bit 0. Phần logic này là mạch tổ hợp.
* (4) Latch CRC result: Phần logic lưu lại giá trị CRC sau mỗi lần tính cho từng nhóm 8 bit.

Trong ví dụ này, dữ liệu đầu vào là 32 bit, việc tính toán sẽ lặp trong 4 chu kỳ, mỗi chu kỳ tính toán trên 8 bit dữ liệu. Kết quả sẽ có ở thanh ghi CRC, *crcSeq*, ở chu kỳ thứ 5.

Các bạn hãy chú ý một số điểm khác biệt so với mạch nguyên lý  tính CRC song song toàn phần ở [bài 3](http://nguyenquanicd.blogspot.com/2019/08/crc-bai-3-bo-tinh-crc-song-song.html" \t "_blank) như sau:

* Thanh ghi lưu dữ liệu ngõ vào là một thanh ghi dịch chứ không phải chỉ là thanh ghi lưu dữ liệu vì nó cần dịch tới nhóm bit tiếp theo cho mỗi lần tính
* Trước từng nhóm bit được ráp thành chuỗi đầu vào cho mạch tính song song, nó được XOR với thanh ghi chứa giá trị CRC hiện tại
* Logic thanh ghi chốt giá trị CRC, crcSeq, có thêm một MUX điều khiển bởi *ctrlEn*để xóa thanh ghi này về 0 cho mỗi lần tính CRC mới.

|  |
| --- |
|  |
| Hình 3: Thanh ghi dịch lưu dữ liệu ngõ vào |

Thanh ghi dịch *dataInReg* lấy dữ liệu ngõ vào mới khi *ctrlEn=1* và dịch trái *CRC\_WIDTH*bit khi tính toán, ứng với *crcReady=0*.

|  |
| --- |
|  |
| Hình 4: Bit báo trạng thái crcReady và bộ đếm tính CRC |

Bộ đếm shiftCounter sẽ giám sát số chu kỳ tính CRC, bộ đến có số bit *COUNTERW*bằng log2 của độ rộng bit dữ liệu chia cho độ rộng chuỗi CRC. Số chu kỳ tính toán CRC *CALNUM* bằng độ rộng bit dữ liệu chia cho độ rộng chuỗi CRC. Nếu số bit dữ liệu là 32, chuỗi CRC là 8 bit thì số chu kỳ tính toán là 32/8=4 và độ rộng bộ đếm là log2(4)=2.  
Bit trạng thái *crcReady*sẽ luôn bằng 1 khi bộ tính CRC sẵn sàng. Khi *ctrlEn=1*, quá trình tính toán bắt đầu, *crcReady*bị xóa về 0. Nó được thiết lập lại mức 1 khi giá trị bộ đếm giá sát chu kỳ tính CRC đã đếm đủ số chu kỳ cần tính, *setReady=1*.

**3) Kết quả mô phỏng**

Một kết quả mô phỏng cho bộ tính CRC từng phần thể hiện sau đây:

* DWIDTH=32
* CRC\_WIDTH=8
* Đa thức sinh sử dụng là x^8+x^2+x^1+1 ứng với giá trị GenPoly=8'h07 trong biểu diễn đa thức sinh thông thường.

|  |
| --- |
| [A screenshot of a computer program  Description automatically generated](https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEie4TLYNm2U6WpIBPtIAaa_lcwmUc96Zfye_sulBVKgadcxidtDYmNZsSAoWS-bwK3iF3lBzSgL3rYHa3CLapYh2qth51ry2VVZmb65h3MX6sC6Hm71uO8LPdTrYV9-OOtdeDCjB1r6OfPC/s1600/image.png) |
| Hình 5: Một waveform của bộ tính CRC từng phần với cấu hình chuỗi CRC là 8 bit và dữ liệu đầu vào là 32 bit |

Lưu ý, bộ tính CRC này chỉ hỗ trợ việc tính toán CRC với số bit dữ liệu cần tính chia hết cho số bit CRC, DWIDTH/CRC\_WIDTH là một số nguyên dương.  
Một cách khác để thực hiện bộ tính CRC từng phần là thay thế logic mạch tính CRC song song bằng một RAM/ROM chứa sẵn giá trị CRC của tất cả các nhóm bit. Trong ví dụ này, nhóm bit có độ dài là 8 thì cần một bộ nhớ chứa 2^8=256 giá trị. Mỗi lần tính, 8 bit dữ liệu sau cổng XOR ở bước tính (2) sẽ là địa chỉ để chọn ô chứa giá trị CRC tương ứng từ bộ nhớ RAM/ROM. Cách làm này giống như việc tra bảng sự thật (lookup table). Để làm được cách này, tất cả các giá trị có thể từ 8'h00 đến 8'hff phải được tính toán trước và lưu trong RAM/ROM trước khi bộ tính CRC hoạt động.

|  |
| --- |
|  |
| Hình 6: Một cách thực hiện bộ tính CRC song song từng phần khác, trường hợp CRC\_WIDTH=8 |

*Web tính CRC onlline dùng để kiểm chứng kết quả:*  
<https://crccalc.com/>  
  
*Dữ liệu có thể download:*  
[Source code trên Github](https://github.com/nguyenquanicd/crcCalculator)