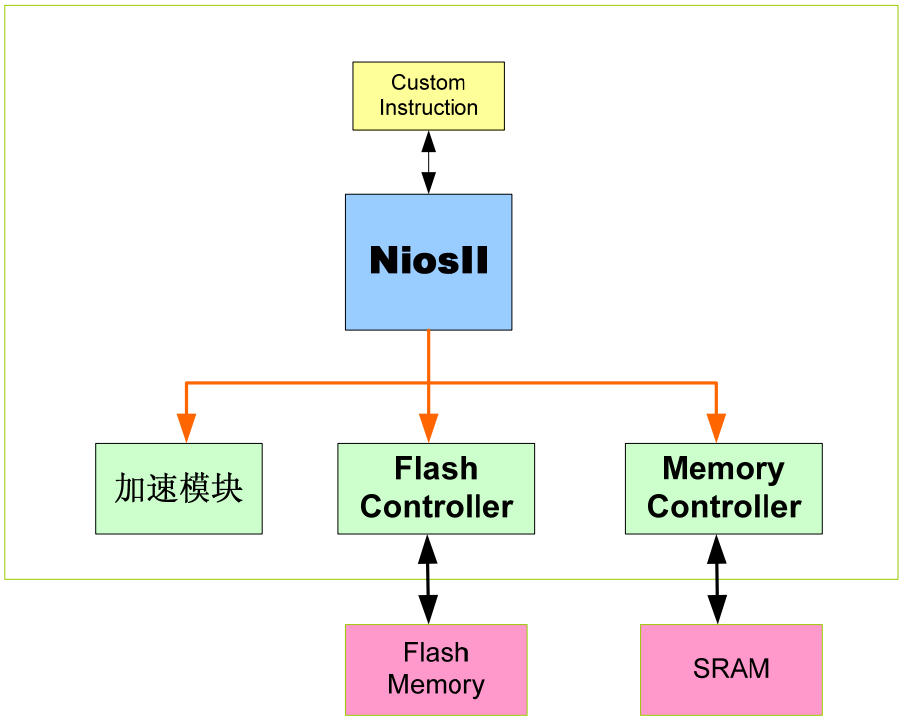
啥是 DSP

- 系統的問題，零零碎碎說了那麼些，最終的目的，其實就是要把積木搭起來，而我們之前說的都是一些工具，我們組建一個系統可以使用的一些資源。看看我們是否可以根據我們對這些積木的了解來組建一個最炫的系統出來。我們首先看看我們有些什麼好了。
-

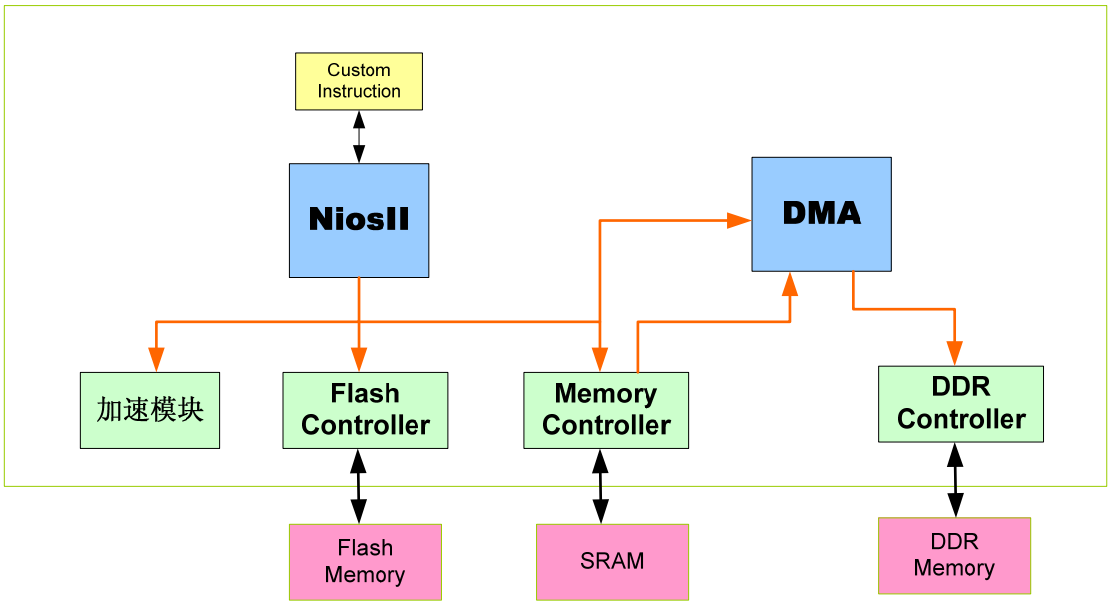
NiosII: 牛死 or 死牛

NiosII 其實是個很好玩的東西，他可以讓你的系統牛死，但是也很容易就變成了死牛。NiosII 作爲一個軟核，好處是可以很容易的實現多核系統，壞處就是他的速度不夠快，一百多兆的速度，顯然和什麼 ARM 啦什麼的沒辦法比了。但是問題是，NiosII 是在 FPGA 中實現的，所以他可以使用非常多的加速模式，使得這頭死牛變得牛死。作爲 CPU，它本身的控制，靈活性是毋庸置疑的，所以我們需要在操作速度上面做文章。把一個幾十，甚至幾百行的軟體代碼，變成硬體，成爲一條指令。對了，我們可以使用自定義指令來實現。比如一個比較龐大的複雜運算，而輸入和輸出又相對單純。這當然是一種模式，這種模式相對來說是最簡單的，因爲軟體裡面只是調用了一條指令而已。但是這種模式有一個小缺點，當你的運算非常大的時候，CPU 不得不停下來等著你完成計算它才會繼續下去。針對這種狀況，我們可選擇使用一個 component 來加速。就是把加速硬體模塊做成一個 component，然後用 Avalon-MM 和它進行連接。這樣在這個模塊進行運算的時候，CPU 並不會暫停，而是不斷的繼續下面的程式。最後 CPU 來看一下結果就好了。但是壞處是，會佔據一段時間的 CPU 上的 avalon-mm 端口。硬體加速的效果是可以驚人的，完全可以做到讓一條死牛，變成真正的牛死。



數據搬運：

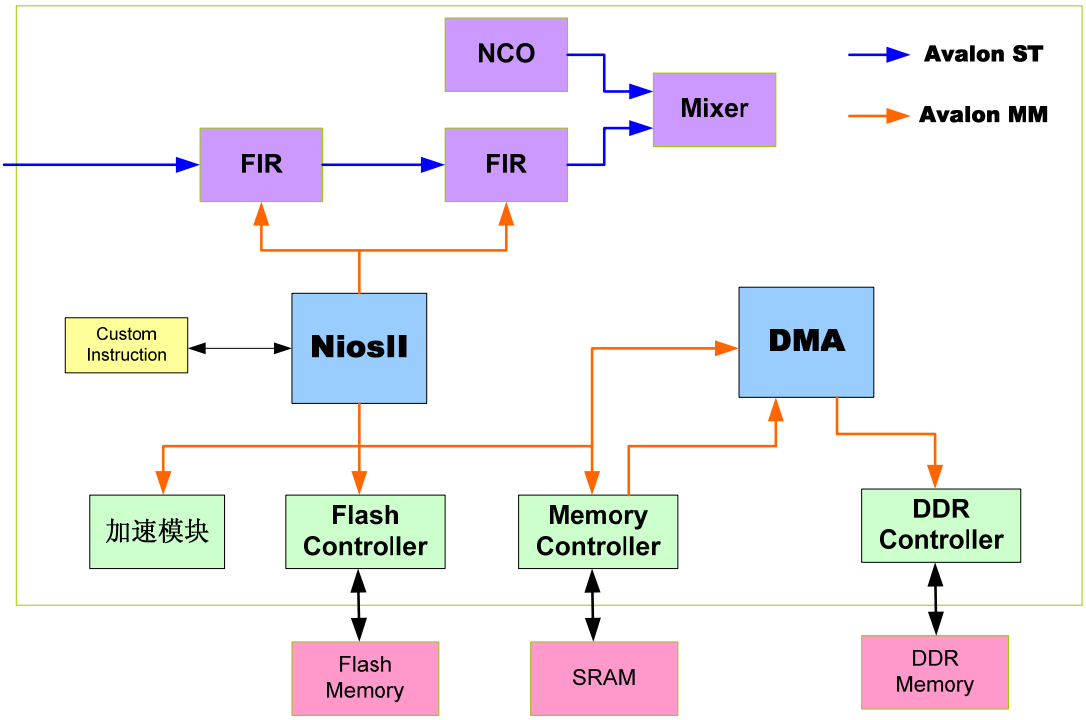
現下又帶來一個新的問題，如果我們需要搬運很多的數據，從一個地方到另外一個地方，怎么辦呢？透過 NiosII 可能會太慢了，同時這樣的事情，是不是太過無聊了一些，用 CPU 來做大材小用了。那麼這個時候，我們可以用一個叫 DMA 的長工來搬箱子。



高吞吐量：

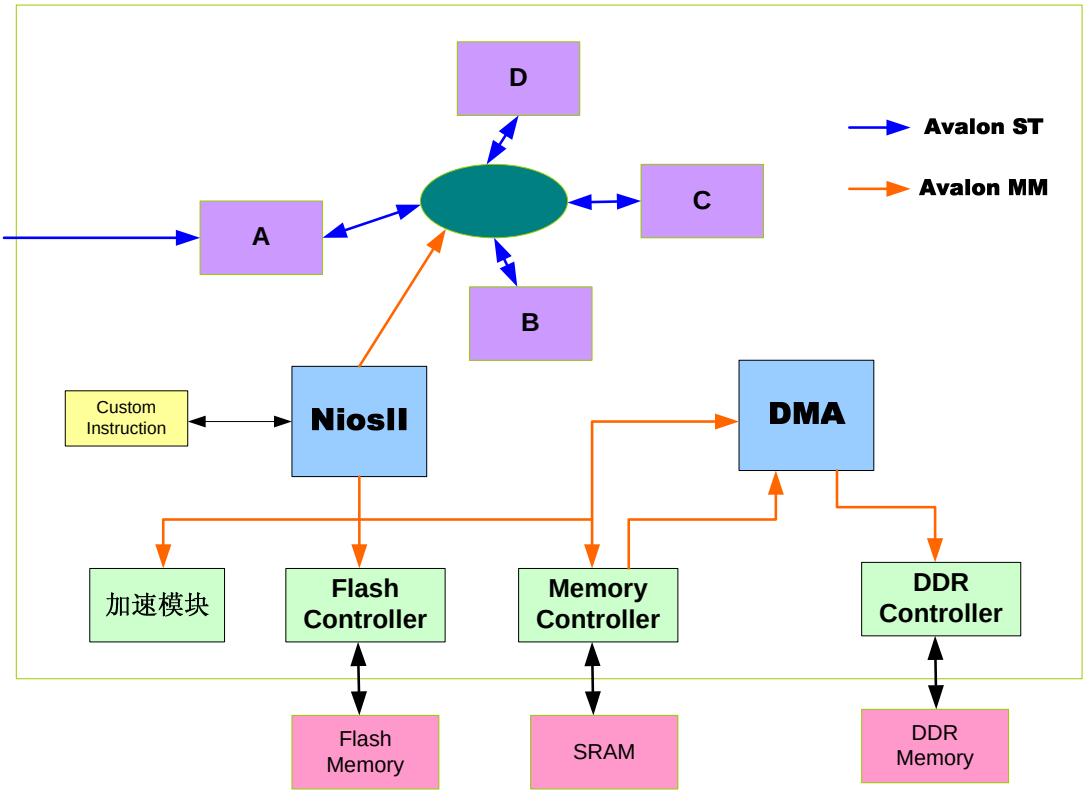
現下要求更高了些了，就是我對數據的計算量要求非常大，數據又是源源不斷的進來了。我們必須很快的處理數據，同時可以做相應的控制。那麼我們就引用了 Avalon-ST 來提升數據

的傳輸速度：



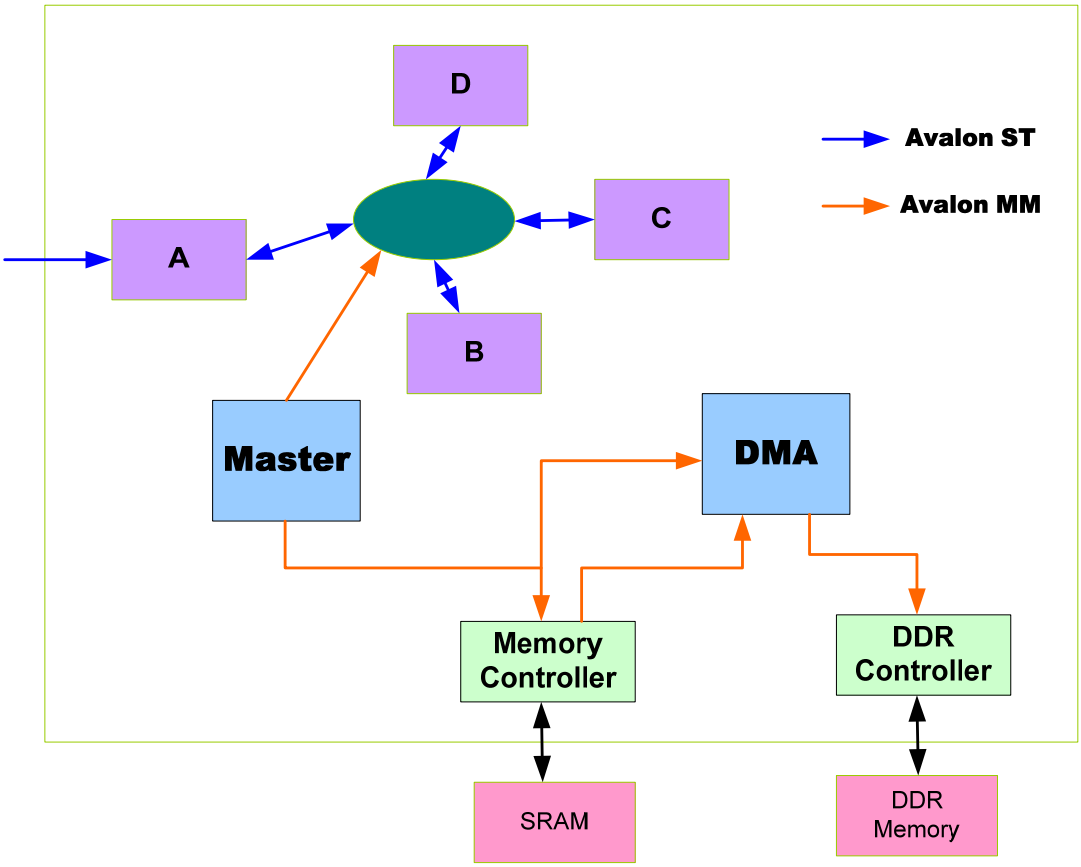
我需要更大的靈活度：

系統又提意見了，這樣的設計速度是有了，可是沒有靈活性啊。針對不同的協議，我可以需要的操作不一樣呀，你不能這么死的連在一起。好像曹操在赤壁一樣，那一把火過來，不全完了嗎？好吧，那麼我們可以做個橋，作爲各個操作模塊之間的控制。它可以根據狀況來控制整個處理的流程。而每個處理模塊也只是和他進行交流，所有的模塊從它獲得數據，然後把數據轉回。



一定要 NiosII 嗎？

不一定。在系統中，沒有任何一樣東西是必須的。只有有用的東西，沒有一定的東西。而去掉一個 NiosII 的好處是可以節省很多資源，包括 FPGA 內的，和外部的。NiosII 沒什麼特別的能力，就是一個大型的狀態機，或者說一個 Master 而已。 當然可以把它拿掉，照樣還是一個 SOPC 系統



所以，系統這個東西，沒有最好的，只有最適合的。你在了解自己的應用的基礎上，選擇你覺得最適合的方案。幸運的是 SOPC 這個平台，已經提供了你所需要的基礎建設。一些具體的模塊當然需要自己去做的。

大家一起來搭積木好了。