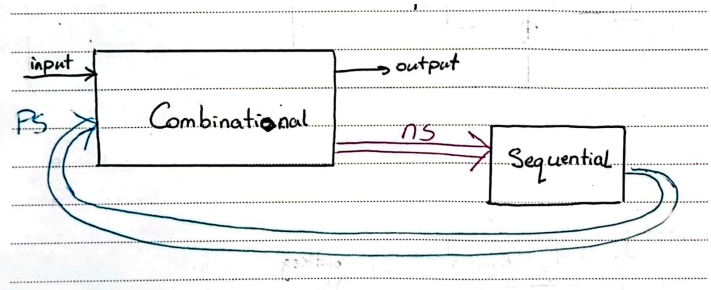
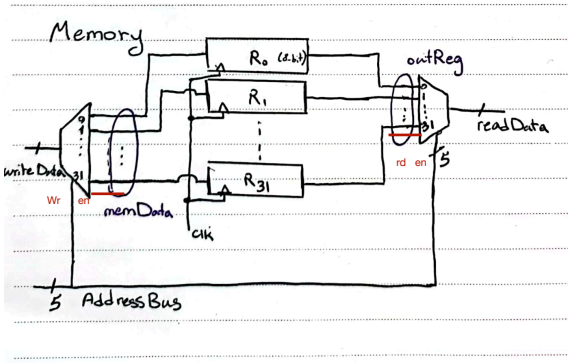
ریحانه حاجبی ۸۱۰۱۰۰۱۱۶ صفورا علوی‌پناه ۸۱۰۱۰۰۲۵۴

برای طراحی bubble sorter با الگوریتم داده شده در صورت پروژه به روش هافمن ، دو بخش combinational و sequentional داریم . و در طراحی آن به بخش datapath و controller نیاز داریم .



بخش datapath شامل موارد زیر است:



۱. Memory برای ذخیره ی اعدادی که باید مرتب شوند. که در این مموری نوشتن به صورت سنکرون و خواندن به صورت آسنکرون انجام می‌شود. در نتیجه برای نوشتن روی رجیسترهای حافظه باید هر بار علاوه بر فعال کردن سیگنال wr روی لبه ی بالا رونده ی کلاک نیز باشیم. در حالی که برای خواندن هر لحظه که enable mux یا همان rd فعال شود data روی خروجی نمایش داده می‌شود و نیازی به کلاک ندارد.

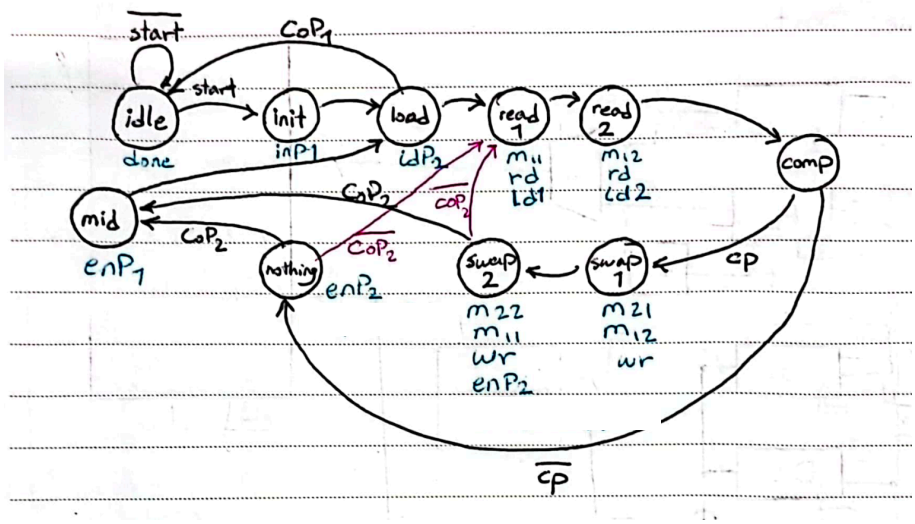
۲. دو counter برای اشاره کردن برای خانه های حافظه به صورت متوالی که یکی معادل حلقه for اولیه در الگوریتم بوده و بعد از هر بار به آخر رسیدن اولی یکی زیاد می‌شود. و شمارنده ی دومی هربار باید با مقدار اولیه ای معادل مقدار اولیه counter اول به علاوه ی یک load شود .

۳. دو رجیستر برای ذخیره ی اطلاعات (مقادیر خانه های حافظه) خوانده شده برای مقایسه نیاز است. که مجدد در دو کلاک متوالی load می‌شوند.

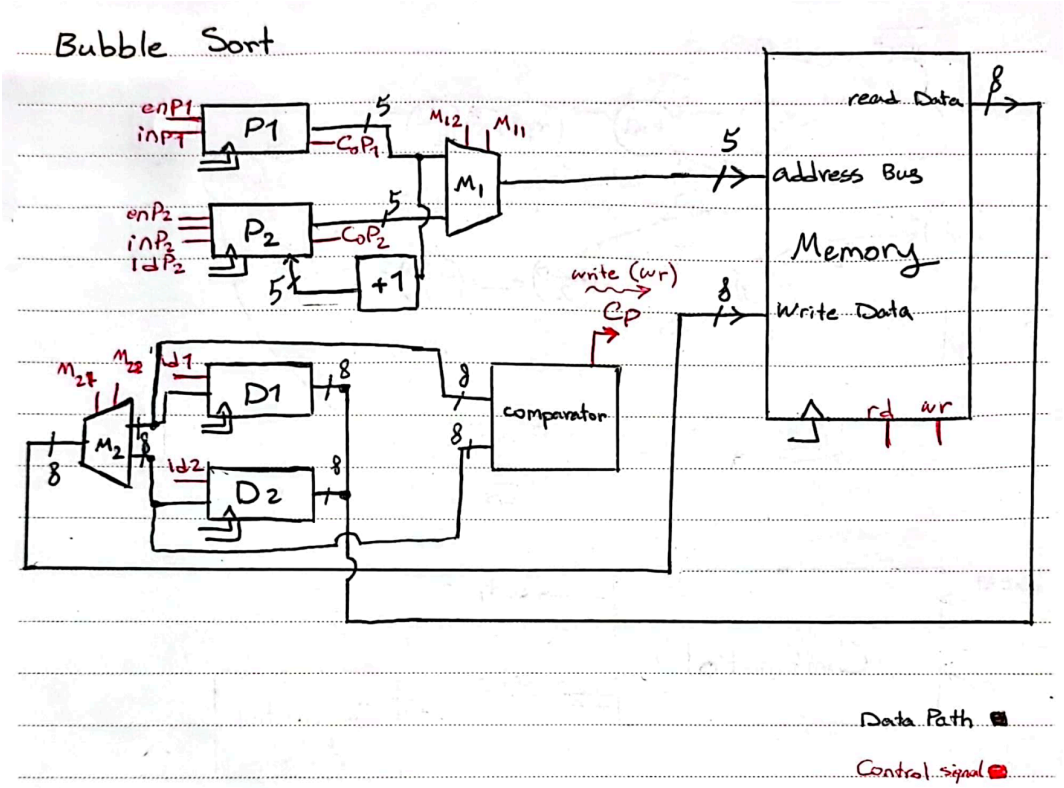
۴. Comparator برای مقایسه ی مقادیر ذخیره شده در رجیستر ها ، که اگر عدد با index کوچک تر مقداری کوچک تر نداشت با سیگنال cp که به واحد کنترل می‌دهد عملیات swap انجام می‌شود.

۵. دو multiplexer برای خواندن و نوشتن متوالی دو خانه ی حافظه از memory .

بخش control unit با state machine زیر به ترتیب دو عدد می‌خواند ، مقایسه می‌کند و در صورت نیاز swap می‌کند ، و در غیر این صورت به state مربوط به خواندن دو خانه ی بعدی می‌رود.



* اصلاحیه شکل: state nothing با صفر بودن coP2 به read 2 می‌رود.



* ما این datapath را با اندکی تفاوت از سودو کد مطرح شده در صورت سوال طراحی کردیم به صورتی که در حلقه ی دوم for به جای شروع از indx2 = indx1 از indx2 = indx1 + 1 شروع میکنیم. که هر خانه با خودش مقایسه نشود . درنتیجه به یک incrementer نیز نیاز داشتیم.

ما کد مربوط به memory را به گونه ای نوشتیم که اعداد را از یک فایل txt خوانده و حاصل مرتب شده را در همان فایل می‌نویسد. درنتیجه نیازی به نوشتن اطلاعات در testbench نداریم.

شکل موج: