دانشگاه صنعتی شریف

دانشکده مهندسی کامپیوتر

پایان نامه کارشناسی ارشد

گرایش معماری سیستمهای کامپیوتری

طرّاحی و پیاده سازی یک مودم ADSL

ابوالفضل شمس

استاد راهنما:

دکتر محمّد تقی منظوری

بهمن ماه ۱۳۸۲

به نام خدا

دانشگاه صنعتی شریف

دانشکده مهندسی کامپیوتر

رساله کارشناسی ارشد

عنوان:.....

نگارش:.....

کمیته ممتحنین:

استاد راهنما:..... **امضاء**.....

استادمشاور:..... **امضاء**.....

استاد مدعو:..... **امضاء**.....

تاریخ:.....

چکیده:

این پایان قسمت دوّم از پروژه "طرّاحی و پیاده سازی یک مودم ADSL" میباشد. در قسمت اوّل که بصورت پایان نامه مجزّا تعریف و پیاده شده است، طیف کانال twisted pair که رسانه انتقال تکنولوژی ADSL است، بررسی و مدلسازی شده است. در این قسمت به طرّاحی و پیاده سازی مودم ADSL پرداخته میشود.

در این پایان نامه یک مودم ADSL طرّاحی و پیاده میشود. علیرغم اینکه در proposal پروژه فقط پیاده سازی وظایف بخش ارسال مورد نظر بود، بمنظور انجام یک کار اضافه بخش دریافت هم پیاده شده است. میتوان گفت که در این پروژه یک مودم ADSL که در تأسیسات مشتری قرار میگیرد، بطور کامل پیاده شده است.

در این پروژه پیاده سازی بخش نرم افزاری بر روی TMS320c6711 DSK در محیط CCS2 انجام گرفته است و با کمک گرفتن وسیع از امکانات پردازنده مثل واحدهای EDMA، McBSP، EMIF و....که در جای مناسب توضیح داده میشود، سعی شده است تا آنجا که ممکن است پیاده سازی بنحو بهینه تری صورت گیرد. همچنین از امکانات CCS2 مانند کتابخانه های غنی CSL ، DSPLIB استفاده شده است.

همچنین طرّاحی بخش آنالوگ مورد بحث قرار گرفته است. این بخش را با ذکر یک شمای کلّی شروع میکنیم و در نهایت به طرح کامل واسط آنالوگ با ذکر IC های مربوطه و حتی پایه هایشان میرسیم.

کلید واژه ها: مودم ADSL، پردازنده های DSP، محیط CCS

فهرست مطالب

فهرست شکلها

مقدّمه)DSL و اهمّیت مدلسازی کانال در آن:

در اثر رشد و گسترش ارتباطات شبکه ای نیاز به انتقال سریع اطّلاعات مسئله ایست که روز به روز مورد توجّه بیشتر قرار میگیرد. بدین منظور تکنولوژیهای متعدّدی عرضه شده است که DSL یکی ازجذّابترین آنها است. ویژگی بسیار مطلوبی که این تکنولوژی را سرآمد همگنان خود ساخته این است که رسانه انتقال در اینجا خطوط Twisted pair می باشد.یعنی همان خطوطی که برای انتقال صوت تلفنی مورد استفاده قرار میگیرد. هنر DSL نیز همین جا ظاهر میشود. در اینجا میتوانیم با انجام تغییرات بسیار جزئی در سیستم انتقال تلفن، داده را هم انتقال دهیم. باین ترتیب دیگر لازم نخواهد بود تأسیسات مجزّا برای انتقال داده ایجاد کنیم و در هزینه فوق العاده صرفه جویی میشود.DSL توانسته است با استفاده از پهنای باندی که برای انتقال تلفنی استفاده نمیشود، داده را با سرعت چند مگابیت بر ثانیه انتقال دهد و بعلاوه در این کار هیچ مزاحمتی برای انتقال صوت تلفنی هم ایجاد نمیکند و حتّی میتواند انتقال داده و صوت همزمان هم انجام گیرد.

امّا درحالتیکه مودمهای dial up، نمیتوانند بیشتر از چندده کیلو بیت داده را انتقال دهندDSL چگونه توانسته است امکان انتقال با این سرعت زیاد فراهم کند؟ پاسخ این سؤال را براساس تدابیر انجام شده در تکنولوژی ADSL، عضوبرجسته خانواده DSL میدهیم و البته برای سایراعضا نیز بطرزی مشابه نتیجه میشود.

در ADSL پهنای باند بین ۲۰k تا حوالی ۱M ، برای انتقال داده اختصاص یافته است. امّا همانطور که میدانیم در این فرکانسها سیم Twisted pair تضعیف فوق العاده شدیدی دارد و بدتر اینکه همانطور که بعداً خواهیم دید بواسطه قرار گرفتن سیمها در کنار هم نویزهایی ایجاد میشوند که در این فرکانسها مقادیر بسیار قابل توجّهی دارند. مقابله با تضعیف که بدون تغییر رسانه ممکن نیست. چون هدف اصلی ما این بوده که از همان خطوط Twisted pair استفاده کنیم. پس در اینجا سعی میشود با نویز مقابله شود. ایده کار هم این است که طیف فرکانسی به چندین بخش تقسیم میشود. سعی میشود از آن بخشهایی که نسبت سیگنال به نویزشان کمتر است، داده کمتری عبور دهیم و از بخشهای با نسبت بالاتر داده بیشتر. میتوان گفت که در اینجا داریم بنحوی اثر نویز را خنثی میکنیم. امّا آیا این تدبیر میتواند مؤثر واقع شود؟ و آیا میتوانیم داده را با سرعت چند مگا هرتز عبور دهیم.

در بخش اوّل این پروژه که همانطور که گفته شد تحت پایان نامه ای مجزّاً تدوین شده است نشان داده شده که این امر ممکن است. بدین منظور در آنجا مدلی برای کانال بدست آورده شده و نشان داده شده حتّی با فرض بدترین مقدار نویز میتوان به این سرعت دست یافت.

امّا در این پایان نامه کار اصلی انجام خواهد شد و به طرّاحی و پیاده سازی مودم خواهیم پرداخت.ساختار این پایان نامه بصورت زیر است:

در بخش اوّل ساختار ADSL، توضیح داده خواهد شد و بخشهای مختلف آن آشنا میشویم. ساختار ارائه شده ساختاری نسبتاً کامل است.

بخش دوّم مروری بر پردازنده DSP استفاده شده در این پروژه TMS320c6711 و برد پشتیبان آن بنام TMS320c6711 DSK است. بطور مختصر با ساختار این پردازنده آشنا خواهیم شد و تا آنجا که به بحث ما مربوط است این پردازنده و امکانات داخلی آن بررسی خواهد شد. همچنین با CCS2، محیط نرم افزاری مرتبط با این پردازنده آشنا میشویم. CCS2 برنامه هایی که با C معمولی نوشته ایم را به اسمبلی مورد استفاده در

پردازنده های DSP شرکت تگزاس –که TMS320c6711 هم در زمره آنها است– تبدیل میکند. علاوه بر این کار CCS مزایای بسیار مفید دیگری هم دارد. از جمله آنها کتابخانه های متعدد که کارهای ما را علی الخصوص در زمینه کار با بخشهای مختلف پردازنده، بسیار آسان میکند.

در بخش سوم با طراحی AFE آشنا میشویم که داده های دیجیتال تولید شده توسّط پردازنده را به سیگنالهایی تبدیل میکند که مستقیماً میتوانند توسّط کانال تلفنی استفاده شوند. این بخش را با بلوک دیاگرام خیلی کلّی آغاز میکنیم و درنهایت به ساختار کامل AFE با ذکر IC های مورد استفاده و حتّی پایه هایشان می رسیم. شماتیک کامل ORCAD مدار در پایان آن بخش آمده است.

در بخش چهارم که میتوان آنرا مهمترین بخش پایان نامه دانست توضیحات کامل پیرامون نرم افزار نوشته شده خواهد آمد. همانگونه که توضیح داده خواهد شد این نرم افزار در محیط CCS نوشته شده است. در این برنامه از امکانات بسیار مفید CCS که توضیح مختصر پیرامون آنها در بخش پنجم بیان شده، بنحو گسترده استفاده میکند.

و بالاخره در بخش پنجم که بخش پایانی است به نتیجه گیری، بیان مشکلات و پیشنهادات اختصاص یافته است.

۱)تکنولوژی DSL و مودم ADSL:

در تکنولوژی DSL از همان خطوط انتقال تلفن، برای انتقال داده استفاده میشود. جدول۱-۱ انواع مختلف تکنولوژیهای DSL، سرعت پشتیبانی کننده، طول کابل و کاربرد آنها را نشان میدهد. امّا مورد بحث ما در این پروژه مودم ADSL است که از اعضای مهم و مورد توجّه خانواده DSL میباشد بنحویکه در حال حاضر پرفروشترین تکنولوژی DSL بشمار می آید و پیش بینی میشود این برتری خود را تا حدّاقل ۵ سال آینده (۲۰۰۸) حفظ کند.

در ADSL، تقسیم پهنای باند به صورت زیر انجام میشود:

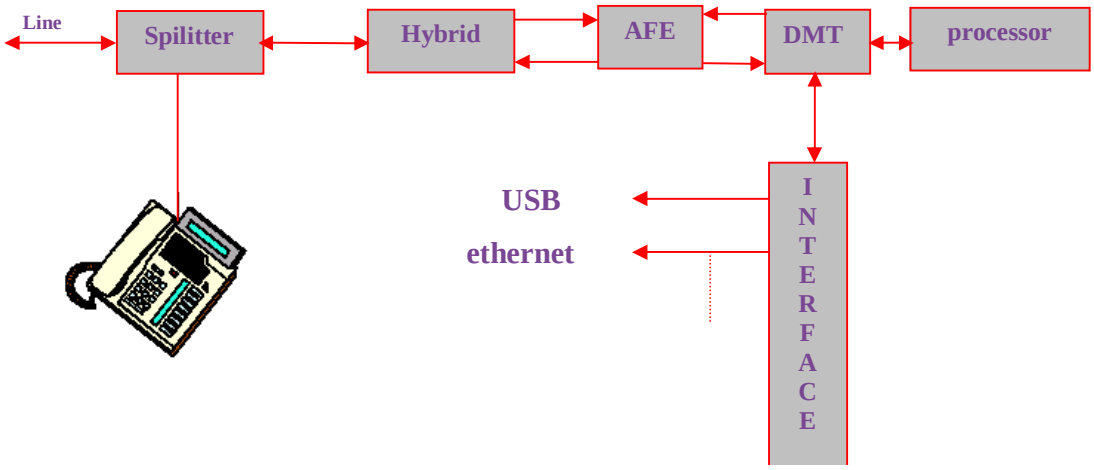
۱- از حوالی ۳۰KHZ تا ۱۳۸KHZ برای upstream (از تأسیسات مشتری به مرکز تلفن)

۲- از حوالی ۱۳۸KHZ تا ۱۱۰۴MHZ برای downstream (انتقال از مرکز تلفن به تأسیسات مشتری)

همانگونه که جدول نشان میدهد هدف ADSL این است که به bitrate بین ۱,۵MHZ تا ۶ MHZ در جهت downstream و معادل ۱۶KHZ تا ۶۴۰KHZ در جهت upstream دست پیدا کند. شکل ۱ طرح کلّی و در عین حال کامل این مودم را نشان میدهد.

جدول ۱-۱) انواع مختلف تکنولوژیهای DSL

DSL Type	Description	Data Rate Downstream; Upstream	Distance Limit	Application
IDSL	ISDN Digital Subscriber Line	128 Kbps	18,000 feet on 24 gauge wire	Similar to the ISDN BRI service but data only (no voice on the same line)
CDSL	Consumer DSL from Rockwell	1 Mbps downstream; less upstream	18,000 feet on 24 gauge wire	Splitter less home and small business service; similar to DSL Lite
DSL Lite (same as G.Lite)	"Splitterless" DSL without the "truck roll"	From 1.544 Mbps to 6 Mbps downstream, depending on the subscribed service	18,000 feet on 24 gauge wire	The standard ADSL; sacrifices speed for not having to install a splitter at the user's home or business
G.Lite (same as DSL Lite)	"Splitterless" DSL without the "truck roll"	From 1.544 Mbps to 6 Mbps , depending on the subscribed service	18,000 feet on 24 gauge wire	The standard ADSL; sacrifices speed for not having to install a splitter at the user's home or business
HDSL	High bit-rate Digital Subscriber Line	1.544 Mbps duplex on two twisted-pair lines; 2.048 Mbps duplex on three twisted-pair lines	12,000 feet on 24 gauge wire	T1/E1 service between server and phone company or within a company; WAN, LAN, server access
SDSL	Symmetric DSL	1.544 Mbps duplex (U.S. and Canada); 2.048 Mbps (Europe) on a single duplex line downstream and upstream	12,000 feet on 24 gauge wire	Same as for HDSL but requiring only one line of twisted-pair
ADSL	Asymmetric Digital Subscriber Line	1.544 to 6.1 Mbps downstream; 16 to 640 Kbps upstream	1.544 Mbps at 18,000 feet; 2.048 Mbps at 16,000 feet; 6.312 Mpbs at 12,000 feet; 8.448 Mbps at 9,000 feet	Used for Internet and Web access, motion video, video on demand, remote LAN access
RADSL	Rate-Adaptive DSL from Westell	Adapted to the line, 640 Kbps to 2.2 Mbps downstream; 272 Kbps to 1.088 Mbps upstream	Not provided	Similar to ADSL
UDSL	Unidirectional DSL proposed by a company in Europe	Not known	Not known	Similar to HDSL
VDSL	Very high Digital Subscriber Line	12.9 to 52.8 Mbps downstream; 1.5 to 2.3 Mbps upstream; 1.6 Mbps to 2.3 Mbps downstream	4,500 feet at 12.96 Mbps; 3,000 feet at 25.82 Mbps; 1,000 feet at 51.84 Mbps	ATM networks; Fiber to the Neighborhood



شکل ۱-۱) ساختار کلّی مودم ADSL

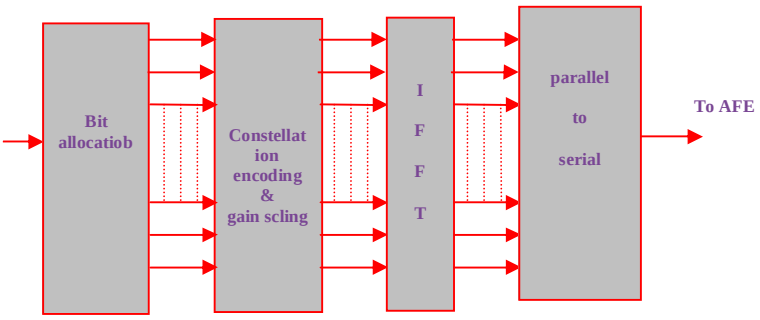
نحوه کار در بخش دریافت سیگنال آنالوگ باینصورت است که:

سیگنال آنالوگ که مشتمل بر صوت تلفنی بعلاوه داده مورد نظر است از طریق خطّ تلفن وارد میشود. در ابتدای کار فیلتر Splitter، طیف صوت را از داده جدا میکند. بدینترتیب کارهایی که مودم بعد از این انجام میدهد هیچ تزاحمی با صوت تلفنی نخواهد داشت.

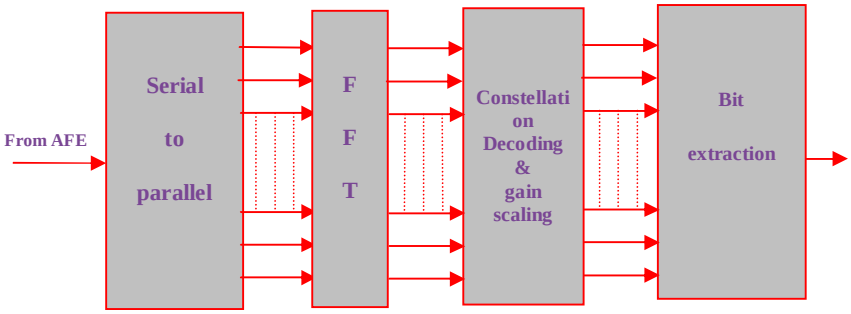
سیستم Hybrid در این شکل برای تبدیل دو سیم به چهار سیم مورد استفاده قرار میگیرد. چون میدانیم اطّلاعات ارسال و دریافت هر دو روی یک خطّ تلفن منتقل میشوند. Hybrid اطّلاعات ارسال و دریافت را از هم جدا میکند و آنها را روی خطوط جداگانه میفرستد.

واحد بعدی AFE است که وظیفه آن تبدیل سیگنال آنالوگ خطّ تلفن به داده دیجیتال مناسب با DSK و بالعکس تبدیل داده دیجیتال به سیگنال آنالوگ خطّ تلفن، میباشد. در نتیجه وجود دو واحد D/A و A/D ضروری بنظر میرسد. بعلاوه در این واحد باید یک سری فیلتر هم قرار داده شود که وظیفه آنها جدا کردن طیف جهت انتقال upstream و downstream میباشد.در این پروژه از Spilitter برای بحثی نخواهد شد. طرّاحی Hybrid و AFE را در بخش بعد بطور کامل بررسی خواهیم کرد.

آخرین واحد، بخش DMT [1] است که مهمترین بخش و واحد اصلی این مودم میباشد. در این واحد مدولاسیون DMT روی داده های دیجیتال انجام میگیرد. همانگونه که قبلاً گفته شد ایده کار ADSL این است که طیف فرکانسی را به بخشهای متعدد تقسیم کند و سپس بر مبنای نسبت سیگنال به نویز به آنها داده اختصاص دهد. امّا هدایت به پهنای باندهای مختلف نوعی مدولاسیون است. یعنی باید در اینجا به دنبال نوعی مدولاسیون بگردیم که بتواند تخصیص داده بر اساس سیگنال به نویز را انجام دهد. مدولاسیون DMT میتواند این کار را برای ما انجام دهد. شکلهای ۱-۲ و ۱-۳ بترتیب ساختار این مودم در بخش ارسال و دریافت را نشان میدهد.



شکل ۱-۲) ساختار DMT در بخش ارسال داده دیجیتال



شکل ۱-۳) ساختار DMT در بخش دریافت از خطّ تلفن

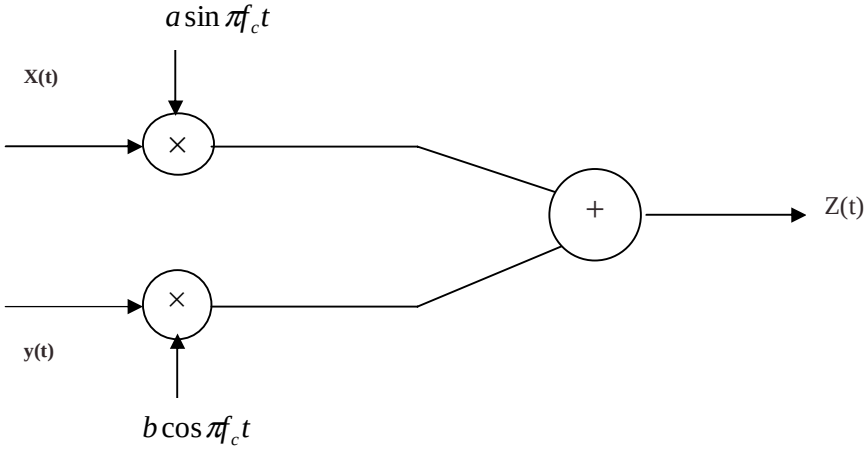
مدولاسیون DMT فی الواقع یک مدولاسیون QAM [2] چندتایی است. بهمین منظور ابتدا توضیحی در مورد QAM ضروری بنظر میرسد.

[1] -Discrete Multi Tone

[2]Quadrature Amplitude Modulation

۱-۱)مدولاسیون QAM:

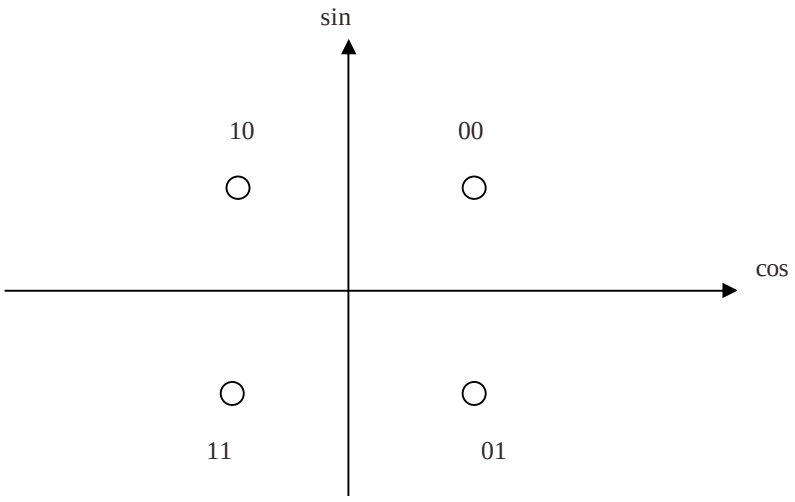
QAM به ما امکان میدهد که دو سیگنال را بکمک یک حامل فرکانسی ارسال کرد. شکل ۱-۴ این مدولاسیون را نشان میدهد:



شکل ۱-۴) ساختار مدولاسیون QAM

برای فهم بهتر ابتدا ساده ترین حالت آن یعنی مدولاسیون دو تایی را در نظر گرفته ایم. فرض کنید میخواهیم داده ۰ و ۱ را با مدولاسیون QAM انتقال دهیم. میتوان نشان داد اگر این دو داده را به آنالوگ تبدیل کنیم و بترتیبی که در شکل بالا نشان شده است انتقال دهیم یعنی در sin و cos ضرب کرده و بصورت یک سیگنال ارسال کنیم در مقصد میتوان این دو را مجدداً بازیافت نمود.

اکنون فرض کنید ۴ داده ۰۰، ۰۱، ۱۰ و ۱۱ داریم. باز هم میتوان نشان داد اگر مطابق شکل ۱-۵ آنها را در سیگنالهای سینوسی و کسینوسی با دامنه هایی که نشان داده ضرب کرده، سپس همه را با هم جمع کنیم میتوانیم در مقصد آنها را بازیابی کنیم. شکل ۱-۵ را یک constellation ۲ تایی یا 2-constellation خوانده میشود.



شکل ۱-۵) constellation دو تایی

بهمین ترتیب میتوان نشان داد که میتوان nتایی constellation هم داشت. در یک n-constellation میتوانیم 2^n داده را انتقال دهیم. بعبارت دیگر میتوانیم داده گسسته [۱] 2^n سطحی را با کمک یک مدولاسیون n QAM تایی با یک حامل فرکانسی انتقال دهیم.

[1] -Discrete

اکنون میتوانیم DMT را بصورت بهتری توضیح دهیم. توضیح خود را بر اساس شکل ۱-۲ دنبال میکنیم. اعدادی که بعنوان مثال می آیند چندان غیر واقعی نیستند و اکثراً در استانداردها باینصورت تعریف شده است.

فرض کنید میخواهیم یک فریم ۱۲۸ بیتی داده را با DMT انتقال دهیم. پهنای باند کانال را بطور فرضی به چندین بخش مساوی مثلاً ۳۲ بخش تقسیم میکنیم. در واحد تخصیص بیت [1] ، این ۱۲۸ بیت در ۳۲ گروه قرار میگیرند به هر گروه i ام، b_i بیت اختصاص داده میشود. سپس این گروهها وارد بخش constellation encoding میشوند. این واحد فی الواقع چیزی جز یک بانک QAM نیست. هر کدام از این گروهها وارد یک واحد QAM میشوند. خروجی هر واحد دو مقدار است که نشان دهنده دامنه کسینوسی و سینوسی میباشد. این دو مقدار را به صورت یک عدد مختلط فرض میکنند تا انجام محاسبات روی آنها راحتتر باشد. پس با اعداد مفروض در اینجا خروجی واحد constellation encoding ، ۳۲ مقدار مختلط میباشد.

سپس این ۳۲ مقدار وارد یک بانک قیلتر IFFT میشود. بکمک این بانک فیلتر است که ۳۲ بخش مختلف در پهنای باندهای مختلف قرار میگیرند. فی الواقع IFFT نقش هدایت طیفی را بر عهده دارد. وروردی IFFT، ۳۲ مقدار مختلط است. امّا چون بهتر است در خروجی فقط مقادیر حقیقی داشته باشیم که کارمان راحتتر باشد مزدوج این ورودیها را هم به ورودیها اضافه میکنیم و در نتیجه یک IFFT ۶۴ نقطه ای میگیریم.

همانطور که ملاحظه شد DMT میتواند هر دو وظیفه هدایت فرکانسی و تخصیص داده بر مبنای SNR را انجام دهد. تخصیص داده بر مبنای SNR در بخش تخصیص بیت انجام میشود. چون وقتی یک بخش شامل بیتهای کمتری باشد مثل این است که داده های کمتری داشته باشد.

در ادامه بحث به چند نکته در مورد این واحدها اشاره خواهد شد که شامل بررسی نحوه تخصیص بیت، استاندارد constellation encoding و اضافه کردن امکانات مقابله با خطا خواهد بود.

[1] -Bit allocation

۱-۲-۱) واحد تخصیص بیت:

طرز کار این واحد بیان شد. امّا نکته ای که باقی مانده است این است که تخصیص بیت بر مبنای SNR چگونه انجام میگیرد.

جهت تخصیص بیت الگوریتمهای متعدّدی ارائه شده اند که به الگوریتمهای bit loading یا bit allocation معروفند. در اینجا به بررسی تئوری نحوه تخصیص بیت نمی پردازیم. تنها به ذکر نتایج یکی از روشهای پراستفاده در اینجا اشاره میکنیم. میتوان نشان داد که b_i، تعداد بیتهای تخصیص داده شده به واحد i ام از فرمول زیر بدست می آید:

$$b_i = \log_2(1+\frac{SNR_i}{\Gamma}) \qquad (۴-۱$$

Γ، که gap خوانده میشود مقدار ثابتی برای همه زیرکانالهااست که بر اساس میانگین هندسی SNR واحدها، تعداد کلّ بیتهای تخصیصی و تعداد واحدها بدست می آید. برای QAM این مقدار حوالی ۱۰db بدست می آید. با داشتن طیف کانال و نویزها که در فصول گذشته بطور دقیق مدل شدند و با توجّه به فرمول استاندارد شده طیف توان ADSL میتوانیم فرمول b_i را براحتی بدست آوریم. ما بدین منظور یک شبیه سازی بر اساس پارامترهای زیر انجام دادیم:

۱- تعداد disturber ها برابر ۴۹

۲- طول کابل ۱۲۰۰۰ فوت

۳- یک bridge tap هم در نظر گرفته شده

۴- نویز مورد استفاده NEXT. برای لحاظ سایر نویزها ۱/۱۰ مقدار NEXT به NEXT اضافه شده است.

۵- شبیه سازی برای انتقال downstream انجام گرفته است

۶- طول فریم ۲۰۷۳ بیت میباشد.

شکل ۱-۶ نمودار تخصیص بیت را نشان میدهد. همانطور که شکل نشان میدهد در فرکانسهای بالا که SNR زیادتر است بیتهای کمتری اختصاص داده شده است.

بر اساس استانداردهای ANSI و ITU، ماکزیمم تعداد بیتهای تخصیصی ۱۵ بیت میباشد.

شکل ۱-۶) نمودار تخصیص بیت

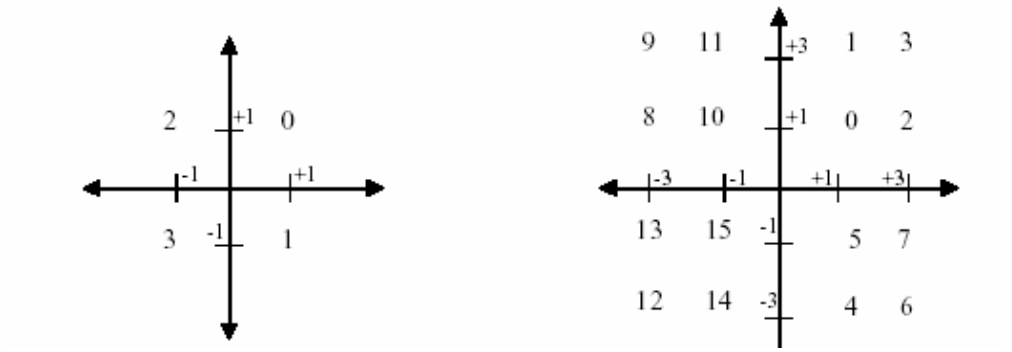
۱-۲-۲) واحد constellation encoding:

روش کار این واحد را توضیح دادیم. در اینجا الگوریتم نحوه تولید constellation بصورتیکه در استانداردهای ANSI 1] و ITU بیان شده توضیح داده خواهد شد. در این استانداردها دامنه کسینوسی و سینوسی اعداد فرد صحیح انتخاب میشود. همچنین بر اساس مقدار b، تعداد بیتهای تخصیصی در QAM برای کدکردن داده $V = \{v_{b-1}, v_{b-2}, ..., v_1, v_0\}$، سه حالت در نظر گرفته شده است:

الف–b زوج باشد. در اینصورت V به صورت زیر کد میشود:

$$x = \{v_{b-1}, v_{b-3}, ..., v_1, 1\}$$
$$y = \{v_{b-2}, v_{b-4}, ..., v_0, 1\} \qquad (۴-۲$$

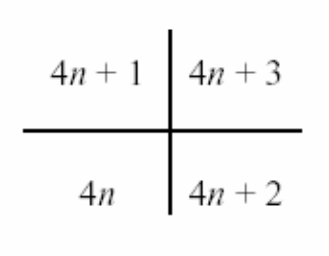
مثلاً بازای b=2 و b=4، constellation بصورت شکل ۱-۷ بدست می آید. در این شکل معادل دهدهی مقادیر نشان داده شده اند.



شکل ۱-۷) constellation بازای b=2 و b=4

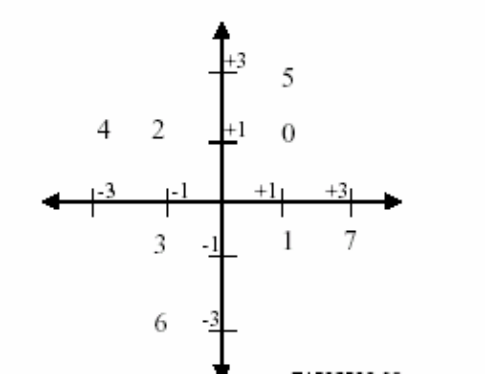
برای سهولت بدست آوردن constellation در این حالت یک قاعده وضع شده است. روش کار باینصورت است که نمودار constellation دو تایی را در نظر میگیریم. آنگاه برای بدست ۴ تایی هر بر چسب با شماره n

را با بلاکی که در شکل ۸-۱ نشان داده شده است، جایگزین میکنیم. همینطور با همین قاعده میتوانیم constellation ۶ تایی را از روی ۴ تایی بدست آوریم و.....



شکل ۱-۸) قاعده بدست آوردن **constellation** های زوج

ب-اگر b=3 باشد، constellation را صریحاً از روی شکل ۱-۹ بدست می آوریم:



شکل ۱-۹)constellation بازای b=3

ج-b فرد و بزرگتر از ۳: در اینصورت x و y از فرمول زیر بدست می آیند:

$$x = \{x_c, x_{c-1}, v_{b-4}, v_{b-6}, ..., v_1, 1\}$$
$$y = \{y_c, y_{c-1}, v_{b-1}, v_{b-3}, ..., v_0, 1\} \qquad (۴-۳$$

در این رابطه مقادیر $c = (b+1)/2$ و $x_c, x_{c-1}, y_c, y_{c-1}$ با توجّه به جدول ۱-۲ بدست می آیند.

جدول ۱-۲) بدست آوردن $x_c, x_{c-1}, y_c, y_{c-1}$

$v_{b-1}, v_{b-2}, \cdots, v_{b-5}$	X_c, X_{c-1}	Y_c, Y_{c-1}
0 0 0 0 0	0 0	0 0
0 0 0 0 1	0 0	0 0
0 0 0 1 0	0 0	0 0
0 0 0 1 1	0 0	0 0
0 0 1 0 0	0 0	1 1
0 0 1 0 1	0 0	1 1
0 0 1 1 0	0 0	1 1
0 0 1 1 1	0 0	1 1
0 1 0 0 0	1 1	0 0
0 1 0 0 1	1 1	0 0
0 1 0 1 0	1 1	0 0
0 1 0 1 1	1 1	0 0
0 1 1 0 0	1 1	1 1
0 1 1 0 1	1 1	1 1
0 1 1 1 0	1 1	1 1
0 1 1 1 1	1 1	1 1
1 0 0 0 0	0 1	0 0
1 0 0 0 1	0 1	0 0
1 0 0 1 0	1 0	0 0
1 0 0 1 1	1 0	0 0
1 0 1 0 0	0 0	0 1
1 0 1 0 1	0 0	1 0
1 0 1 1 0	0 0	0 1
1 0 1 1 1	0 0	1 0
1 1 0 0 0	1 1	0 1
1 1 0 0 1	1 1	1 0
1 1 0 1 0	1 1	0 1
1 1 0 1 1	1 1	1 0
1 1 1 0 0	0 1	1 1
1 1 1 0 1	0 1	1 1
1 1 1 1 0	1 0	1 1
1 1 1 1 1	1 0	1 1

۱-۲-۳) gain scaling:

همانگونه که گفتیم در constellation encoding دامنه سینوس و کسینوس اعداد فرد صحیح انتخاب میشود. امّا هنگام ارسال داده ها به AFE باید دامنه بصورتی باشد که بتواند که مناسب برای ارسال باشد. بهمین منظور بعد از constellation encoding و قبل از واحد IFFT، هر نقطه x+iy در یک مقدار g ضرب میشود. معمولاً تعیین این مقدار در فاز قبل از شروع ارسال توسّط مودمها بدست آورده میشود. مثلاً میتوانند توافق کنند که در ابتدا داده مشخّصی باید دریافت شود. آنگاه مودم طرف مقابل بر اساس مقدار دریافت شده و تفاوت آن با مقدار مفروض gain scaling را بدست می آورد و این اطّلاعات را به مودم دیگر می فرستد.

۱-۲-۴) IFFT:

همانگونه که گفته شد نقش این واحد هدایت طیفی داده است. چون سایز IFFT ، توانی از ۲ است (۶۴) میتوانیم از الگوریتم بهینه butter fly که در درس DSP با آن آشنا شده ایم استفاده کنیم. در اینجا در مورد این الگوریتم توضیحی نمیدهیم.

۱-۳) افزودن امکانات مقابله با خطا:

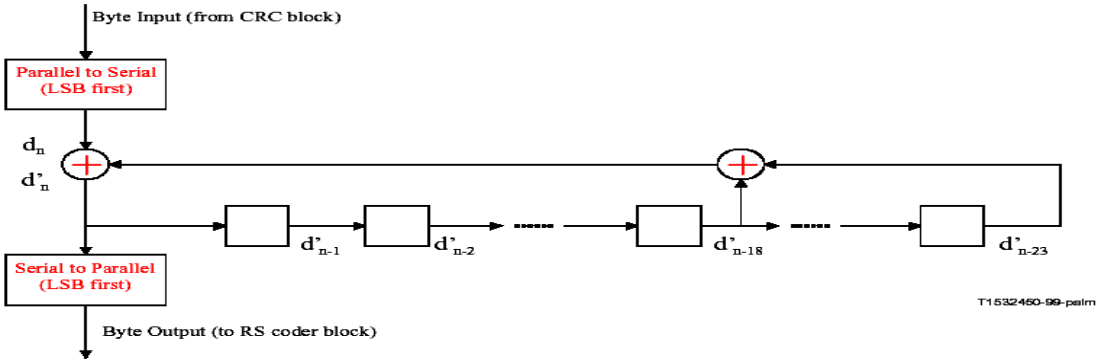
در استانداردهای ANSI و ITU برای افزودن مقاومت در مقابل خطا تمهیداتی اندیشیده شده که در زیر به بعضی از آنها اشاره میشود:

۱-۳-۱) Scrambler:

برای مقابله با رخداد رشته های طولانی ۰ یا ۱ بکار میرود. ایده کارش این است که این رشته ها را با مقادیر شبه تصادفی جایگزین میکند و در گیرنده با unscrambler کردن به داده اصلی دست پیدا میکنیم. در اینجا در مورد تئوری کار توضیح نمی دهیم. در استانداردهای ANSI و ITU این واحد مطابق شکل ۱-۱۰ بدست می آید. فرمول scrambling هم بصورت زیر میباشد:

(۴-۴

$$d_n'(n) = d_n(n) \oplus d_{n-18}'(n) \oplus d_{n-23}'(n)$$



شکل ۱- ۱۰) scrambler

۲-۳-۱) Reed Solomon coding:

برای تشخیص و تصحیح خطا مفید میباشد. روش ساخت آن بدینصورت است که R بایت اضافی

$c_0, c_1, \ldots . c_{R-1}$

به k بایت داده $m_0, m_1, \ldots . m_{k-1}$، اضافه میشوند و بدینصورت N=K+R بایت بدست می آید. روش ساخت این کد از فرمولهای زیر بدست می آید:

$$C(D) = M(D)D^R \bmod uloG(D)$$
$$M(D) = m_0 D^{k-1} + m_1 D^{k-2} + \ldots + m_{k-2} D + m_{k-1} \qquad (۴-۵$$
$$C(D) = c_0 D^{R-1} + c_1 D^{R-2} + \ldots + c_{k-2} D + c_{k-1}$$
$$G(D) = \prod (D + \alpha^i)$$

G(D) چند جمله ای تولید کننده خوانده میشود.

۲-آشنایی با TMS3206711 DSP، TMS3206711 DSK و محیط CCS:

در این بخش پیش نیازهای لازم برای شناخت پردازنده مورد استفاده در پروژه و محیطهای نرم افزاری و کتابخانه ای لازم بیان خواهد شد.

TMS3206711 که منبعداز آن با عنوان DSP6711 یاد میشود یک پردازنده floating point است که توسّط شرکتTexas Instruments (TI)ساخته شده است. مشخّصات کلّی آن بشرح زیر میباشد:

CPU:

۱-دارای ۸ واحد عملیّاتی مستقل شامل ۶ ALU و ۲ ضرب کننده سخت افزاری

۲-۳۲ ثبّات همه منظوره ۳۲ بیتی

۳-MIPS: ۹۰۰ ملیون

۴-Cycle time: ۶,۷ نانوثانیه

۵-انجام ۸ دستورالعمل[1] در ثانیه

۶-۸ بیت برای محافظت از overflow

۷-پشتیبانی از little-endian وbig-endian [2]

[1] Instruction

[2]-در ریزپردازنده های با قابلیّت آدرس دهی بایتی دو استاندارد برای مرتّب کردن data وجود دارد: ۱-little-endian: که در ان بایتها از راست به چپ شماره میگیرند و بیت باارزشتر آدرس بالاتر دارد. ۲-big-endian: ÷که در آن بایتها از چپ براست مرتّب میشوند و در نتیجه بیت با ارزشتر آدرس کوچکتر دارد.

۸-۳۲K کش برای برنامه که L1P نامیده میشود.

۹-۳۲ K کش برای داده که L1D نامیده میشود.

۱۰-۵۱۲K کش برای برنامه و داده بنام L2

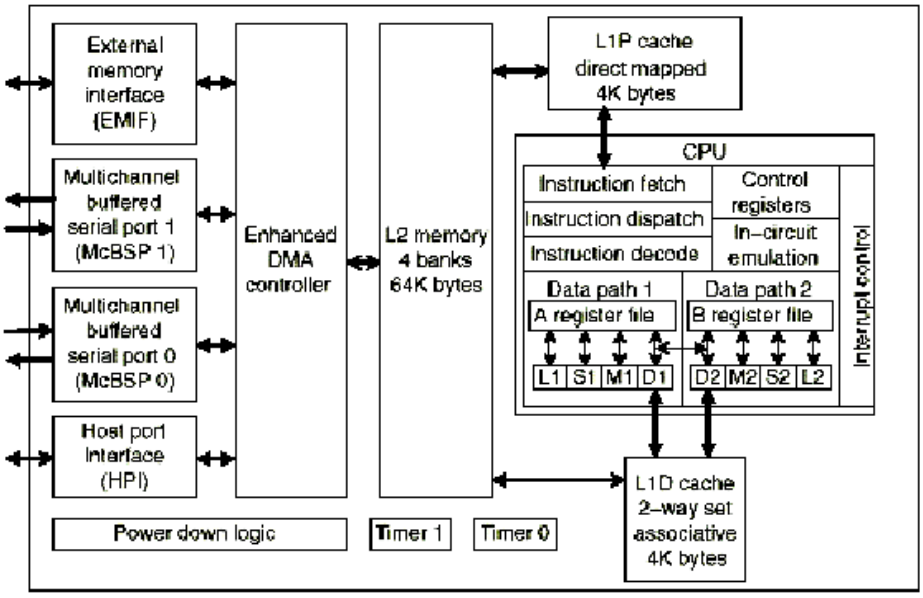
۱۱-EDMA

۱۲-EMIF

۱۳-McBSP

۱۴-Timer

موارد ۱۱ تا ۱۴ بزودی توضیح داده خواهد شد. شکل ۲-۱ ساختار کلّی DSP6711 را نشان میدهد.



شکل ۲-۱) ساختار کلّی DSP6711

اجزای مهم:

ابتدا این اجزاء فهرست وار بیان میشوند. توضیحات کاملتر بزودی خواهد آمد.

۱-EMIF[1]:این وسیله امکان اتّصال حافظه جانبی و وسایل memory map را فراهم میکند.

[1] External Memory Interface

۲-McBSP [1]: شامل دو واحد مستقل بنامهای McBSP0 و McBSP1 است.این واسط امکان ارسال و دریافت سریال داده را فراهم میکند. بکمک این وسیله است که میتوانیم با بخشهای آنالوگ daughter board ارتباط برقرار کنیم.

۳-EDMA [2]: بکمک ۱۶ کانال وقفه موجود در آن امکان انتقال اطّلاعات بدون دخالت CPU فراهم میشود.

۴-دو timer بنامهای timer0 و timer1 که بکمک آنها میتوان زمانبندی انجام داد.

۵-HPI [3]:که امکان انتقال داده میان PC و DSP را فراهم میکند.

DSK TMS320C6711[4]:

TMS320c6711 DSK که من بعد از آن با عنوان DSK6711 یاد خواهد شد یک starter kit ساخت شرکت TI است که کار با DSP6711 را آسانتر میکند. شکل ۲-۲ DSK6711 را نشان میدهد

محتویات :

۱) DSP6711 که توضیح داده شد.

۲) SDRAM ۱۶M ۱۰۰ مگاهرتزی

۳) حافظه Flash ۱۲۸K

۴) پورت memory map I/O ۸ بیتی

۵) کدک صوتی ۱۶ بیتی

۶) کانکتورهای بنامهای J1 و J3 برای ارتباط با daughter board

۷) LED برای نشان دادن وضعیّت روشن شدن، Reset و..

۸) سه LED دیگر که عملکردشان توسّط کاربر قابل برنامه ریزی است

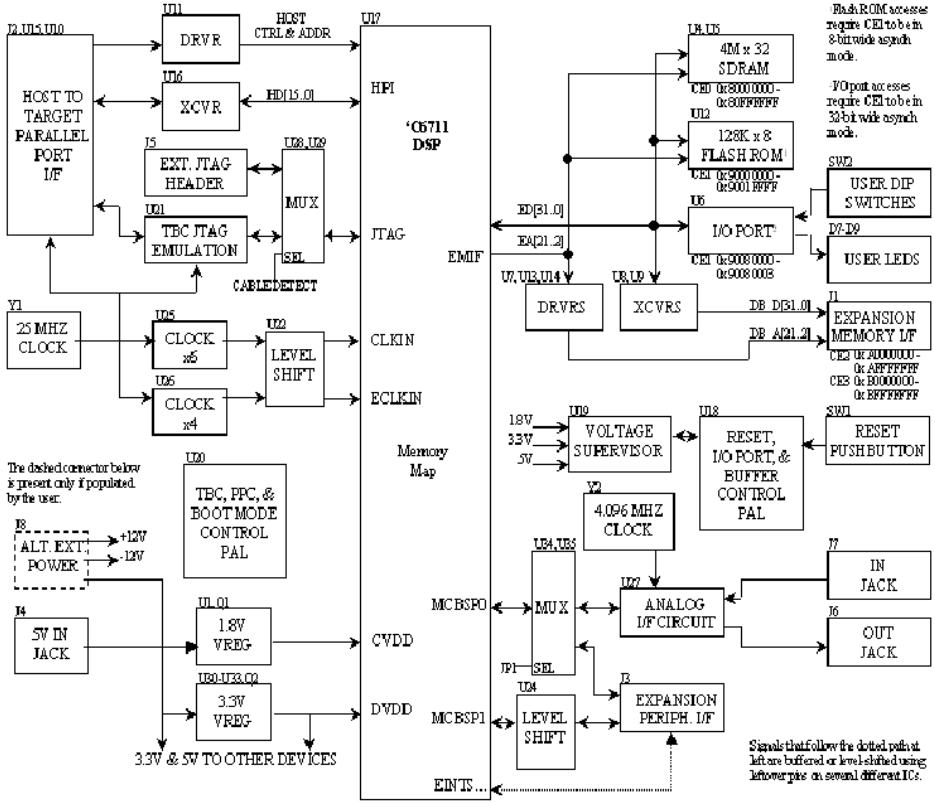
۹) کانکتور IEEE 1284 ۲۵pin بنام J2 برای ارتباط با پورت موازی کامپیوتر

۱۰)کاتکتور منبع تغذیه ۵ ولتی بنام J4 . منبع تغذیه DSK6711 خارجی است.

[1] Multi channel Buffered Serial Port
[2] Enhanced Direct Memory Access
[3] Host Port Interface
[4] DSP Starter Kit

۱۱)کانکتورهای J6 و J7 بترتیب برای خروجی بلندگو ومیکروفن

.

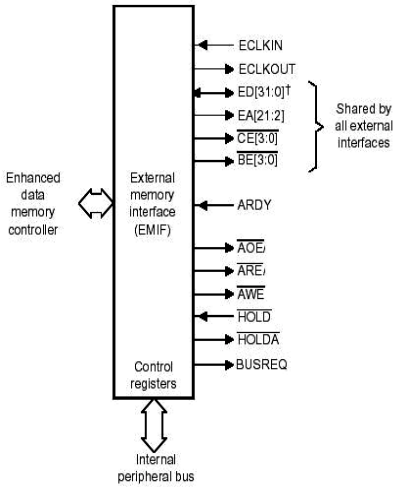


شکل ۲-۲)DSK6711

۲-۱) اجزای مهم DSP6711:

۲-۱-۱)EMIF:

شکل ۲-۳ ساختار EMIF را نشان میدهد.



شکل ۲-۳)ساختار EMIF

اجزاء :

۱-۲۰ بیت برای آدرس دهی حافظه ([EA[21:2)

۲-۳۲ بیت داده([ED[31:0)

۳-قابلیبت آدرس دهی داده های ۸، ۱۶ و ۳۲ بیتی

۴-سیگنالهای کنترلی AWE(Write enable)، ARE(Read enable)،

AOE(Output enable) بمنظور برقراری ارتباط با وسایل جانبی

۵-ARDY: بکمک این پایه میتوانیم به EMIF ،wait cycle اضافه کنیم. در نتیجه ابزارهای جانبی کند هم میتوانند با EMIF ارتباط داشته باشند.

۶-CE[3:0]: برای فعّال کردن فضاهای حافظه ای مختلف. این مورد نیاز به کمی توضیح دارد:

کلّ فضای حافظه ای که توسّط EMIF پشتیبانی میشود، به ۴ بخش مستقل از هم بنامهای CE0 تا CE3 تقسیم شده اند. این اجازه میدهد که مثلاً ۴ وسیله که خطوط آدرس EMIF مشترکی دارند بتوانند بطور مستقل کار کنند.

۷-ثبّاتهای کنترلی: GBLCTL و CE0-4 که توضیح آنها خواهد آمد.

انواع حافظه های قابل پشتیبانی:

EMIF، در DSP6711 قابلیّت ارتباط با حافظه های ROM، SDRAM و SBSRAM دارد. نحوه ارتباط هر یک از این حافظه ها در [۳] و [۴] آمده است. مثلاً این DSP میتواند حدّاکثر ۲۵۶M حافظه SDRAM، شامل ۴ بانک بصورت زیر پشتیبانی کند:

جدول ۲-۱)نحوه پشتیبانی از 256M SDRAM

Bank	width	Depth	Column address	Row adress	Bank select
4	8	8M	A9–A0	A12–A0	A14–A13
			EA11–EA2	EA14–EA2	EA16–EA15

ثبّاتهای مهم:

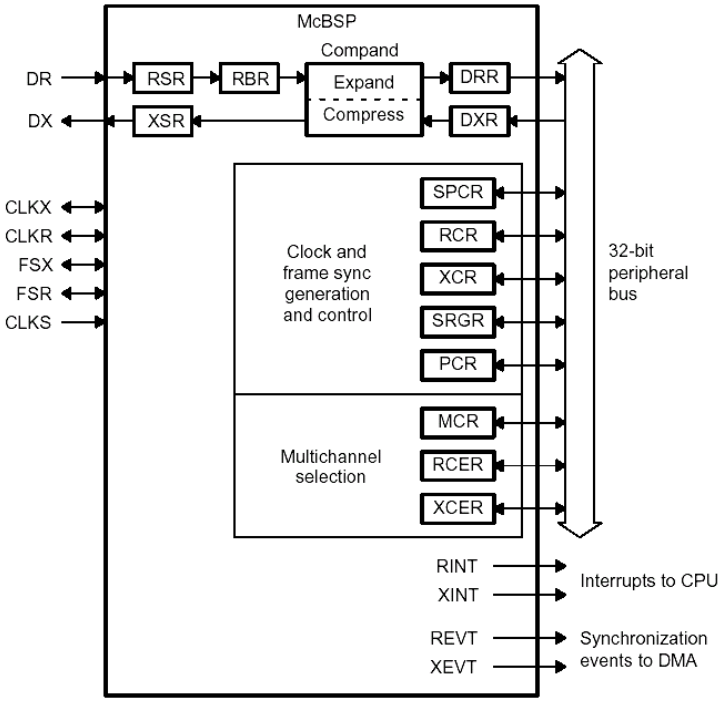
۱- ثبّات GBLCTL: برای پیکربندی کلاک و تشخیص وضعیّت (مثلاً ready بودن یا نبودن) بکار میرود.

۲- ثبّاتهای CE0 تا CE3 برای پیکربندی فضاهای مختلف حافظه ای استفاده میشود.

توضیحات مربوط به نحوه استفاده EMIF در پروژه در بخشهای مناسب خود خواهد آمد.

:McBSP(۲-۱-۲

برای ارسال و دریافت اطّلاعات سریال بکار میرود. دو واحد مستقلّ McBSP بنامهای McBSP0 و McBSP1 وجود دارد. شکل ۲-۴ ساختار McBSP را نشان میدهد:



شکل ۲-۴) ساختار **McBSP**

اجزاء مهم:

۱- DR: پایه ایست که داده سریال به آن وارد میشود

۲- XR: پایه ایست که داده سریال از آن خارج میشود.

۳- CLK X/R:کلاک ارسال/دریافت

۴- FSX/R: سیگنال .Frame Synch برای نشان دادن فرارسیدن زمان ارسال/دریافت

۵- سیگنالهای وقفه: شامل سیگنالهای وقفه به RINT,XINT:CPU و سیگنالهای وقفه به REVT:EDMA و XEVT.

۶- ثبّاتهای کنترلی:SPCR, RCR, XCR, SRGR ,PCR که توضیح داده خواهند شد

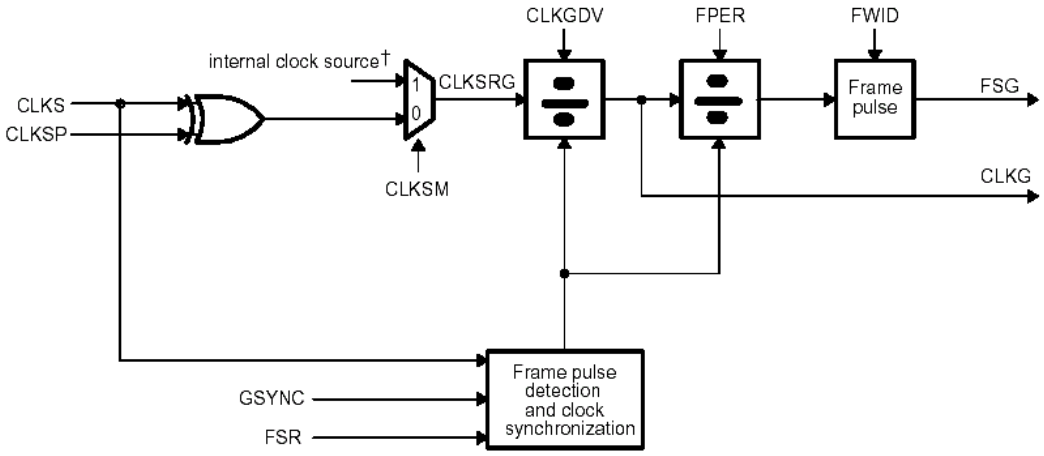
۷- ثبّات RSR که داده سری وارد آن میشود و بطور موازی از آن خارج میشود.

۸- ثبّات XSR:که داده موازی از طرف McBSP وارد آن میشود و به طور سریال از این ثبّات خارج میگردد.

۹- ثبّات DXR: داده ای که قرار است توسّط XSR بصورت سریال تبدیل شود را از CPU دریافت میکند.

۱۰- ثبّات DRR: داده ای که توسّط RSR موازی شده را میگیرد و سپس CPU مقدار را میتواند از آن بخواند.

۱۱- واحدSRG[1]: که دو سیگنال clock و synchronization را تولید میکند. ما بسته به کاربرد میتوانیم از اینها استفاده کنیم یا اینکه McBSP را طوری پیکربندی کنیم که clock را از خارج و توسّط خود ما بگیرد. شکل ۲-۵ ساختار داخلی SRG را نشان میدهد.



شکل ۲-۵) واحد SRG

نحوه ارسال داده سری:

[1] sample rate generator

۱- CPU، داده را در XDR میریزد.

۲- اگر XSR خالی بود محتوای DXR در XSR کپی میشود. در این حالت یک frame synch تولید میشود که به گیرنده (مثلاً وسیله ای روی daughter board) اعلام میدارد که هنگام ارسال داده فرا رسیده. متعاقب این اعلام XSR، داده ها را به صورت سری از پایه DX، خارج میکند.

۳- وقتی DXR کاملاً در XSR کپی شد XRDY=1 میشود و CPU میتواند داده دیگری را بریزد.

نحوه دریافت داده سری:

۱- McBSP با دیدن FSR میفهمد که داده میخواهد بیاید.

۲- سپس داده را از طریق RSR میگیرد.

۳- RSR، داده را در ثبّات بافر RBR میریزد.

۴- محتوی RBR درDRR کپی میشود.بعد ازاین مرحله RRDY=1 میشود و McBSP باز قادر به دریافت داده جدید خواهد بود.

ثبّاتهای کنترلی:

۱- ثبّات SPCR[1]:جهت تنظیمهای عمومی بکار میرود. مثلاً اینکه frame synch ها چگونه تولید شوند برای reset کردن بخشهای مختلف McBSP و..... .

۲- ثبّاتXCR (RCR) : برای تنظیم پارامترهای ارسال (دریافت) بکار میرود. مثلاً اینکه اندازه داده ارسالی(دریافتی) چقدر باشد.در چند مرحله ارسال(دریافت) شود و ....

۳- ثبّات PCR [2]:نحوه پیکربندی پینها را مشخّص میکند. مثلاً اینکه CLKX، CLKR، FSR یا FSX ورودی باشند یا خروجی و....

۴- ثبّات SRGR: وظیفه پیکربندی واحد SRG را برعهده دارد. مثلاً میتواند با تنظیم CLKGDV وFPER که در شکل ۲-۵ ملاحظه میشوند clock و .Frame synch را کنترل کند.

[1] Serial Port Control Register
[2] Pin Control Register

تفصیل کامل مطالبی که در اینجا به طور بسیار فشرده بیان شد و همچنین نمونه هایی از کاربرد McBSP را میتوانید در[۱] و [۵] بیابید.

۲-۱-۳) EDMA:

برای انتقال مستقیم از وسایل جانبی به حافظه، بدون دخالت CPU بکار میرود. در DSP6711، ۱۶ کانال وقفه DMA وجود دارد که مهمترین آنها بقرار زیرند:

۱- ۴ وقفه برای زمانبندی ارسال و دریافت در McBSP0 و McBSP1

۲- ۲ وقفه برای timer0 و timer1

۳- ۴ وقفه جهت وسایل خارجی که با فعّال کردن آنها میتوانند داده را با CPU ردوبدل کنند.

پیکربندی وقفه های EDMA:

EDMA یک حافظه RAM کوچک (2K) بنام PaRAM دارد که اطّلاعات مربوط به پیکربندی وقفه ها باید در آنجا ثبت شود. بازای هر کانال وقفه ۲۴ بایت حافظه اختصاص داده شده است. شکل ۲-۶ محتویات این ۲۴ بایت را نشان میدهد.

31 — 16	15 — 0	
Options (OPT)		Word 0
SRC Address (SRC)		Word 1
Array/frame count (FRMCNT)	Element count (ELECNT)	Word 2
DST address (DST)		Word 3
Array/frame index (FRMIDX)	Element index (ELEIDX)	Word 4
Element count reload (ELERLD)	Link address (LINK)	Word 5

شکل ۲-۶) محتوی یک خانه **PaRAM**

فیلدهای PaRAM:

۱- OPT: در اینجا پیکر بندی مواردی از قبیل تنظیم سایز انتقال، نحوه انتقال و فعّال کردن وقفه خاتمه سرویس بکار میرود. انتقال در EDMA به دو صورت یک بعدی و دو بعدی انجام پذیر است. در انتقال یک بعدی داده ها در بلاکهایی از فریمهای با عناصر[1] یکسان سازماندهی میشوند. در انتقال دو بعدی داده ها در بلاکهایی از آرایه هایی با عناصر یکسان سازماندهی میشوند. جزئیّات این دو مکانیزم در [۳] بتفصیل شرح داده شده است.

منظور از وقفه خاتمه سرویس هم بزودی بیان خواهد شد.

۲- ELECNT: تعداد عناصری که باید منتقل شوند درون این ثبّات قرار میگیرد.

۳- FRMCNT: تعدا فریمها (در انتقال یک بعدی) یا آرایه ها (در انتقال دوبعدی) که باید منتقل شوند.

۴- SRC:آدرس مبدأ انتقال

۵- DST: آدرس مقصد انتقال

۶- ELEIDX: اندیسی که با آدرس فعلی جمع میشود و آدرس لازم برای انتقال بعدی را تولید میکند.

۷- ELERLD: برای ریختن مجدد مقدار ELECNT در انتقال یک بعدی که FRMCNT بیشتر از صفر از بکار میرود. چون در پروژه، ما با انتقال یک بعدی و یک فریمی عادّی کار کرده بودیم این فیلد اصلاً مورد نیاز واقع نشد.

۸- LINK: در DSP6711 بکمک این فیلد میتوانیم به EDMA بگوییم وقتی انتقالی که با فیلدهای قبلی توضیح داده شده پیکربندی شده به اتمام رسید، انتقالی را که پارامترهای آن در آدرس مشخّص شده در LINK، قرار دارند را آغاز کن.

[1] Elements

ثبّاتهای مهم EDMA:

ER-۱ [1]:رخداد وقفه ای که به EDMA داده میشود،بسته به اینکه کدام کانال منبع آن باشد، در بیت مناسبی از این ثبّات ثبت میگردد، حتّی اگر وقفه مورد نظر را بطریقی که بعداً میگوییم غیر فعّال کرده باشیم.

EER-۲ [2]: برای فعّال کردن یا عدم فعّالسازی وقفه ها بکار میرود. با غیرفعّال کردن وقفه مورد نظر دیگر باین وقفه سرویس داده نمیشود. البتّه همانطور که گفته شد رخداد وقفه در ER ثبت میشود.

ECR-۳ [3]:زمانی که یک وقفه در ER ثبت شد و بعبارت دیگر بیت مناسب آن وقفه در ER، set شد، برای پاک کردن این بیت دو راه وجود دارد:

الف-راه خودکار: زمانی که EDMA وقفه ای را پذیرفت بیت متناسب آن در ER بطور خودکار پاک میشود. واضحاً زمانی که وقفه غیر فعّال است این روش کاربرد ندارد چون اصلاً پذیرشی صورت نمیگیرد. در این حالت باید از روش ب استفاده کرد.

ب-از طریق ECR: با set کردن بیت متناسب در ثبّات ECR ، بیت متناظرش در ER پاک میشود.

ESR-۴ [4] : توضیح آن در بخش بعد خواهد آمد.

CIER-۵ [5]: توضیح آن در بخش سرویس خاتمه وقفه خواهد آمد.

CIPR-۶ [6]:توضیح این ثبّات نیز در بخش سرویس خاتمه وقفه خواهد آمد.

نحوه شروع بکار وقفه:

[1] Event Register
[2] Event Enable Register
[3] Event Clear Register
[4] Event Set Register
[5] Channel Interrupt Enable Register
[6] Channel Interrupt Pending Register

شروع بکار وقفه به دو صورت میتواند انجام شود:

۱- مستقیماً از طریق CPU: CPU میتواند با Set کردن بیت متناظر در ثبّات ESR وقفه را مستقیماً راه اندازی کند. در اینجا EER هیچ تأثیری ندارد و صرفنظر از مقدار بیتهای آن وقفه راه اندازی میشود. از این روش میتوان برای راه اندازی نرم افزاری وقفه استفاده کرد.

۲- راه اندازی توسّط رخدادها: به شرط آنکه بیت مناسب در EER ، Set شده باشد، رخدادها مثلاً رخدادهای ناشی از وسایل خارجی میتوانند با فعّال کردن پایه وقفه مناسب، به EDMA وقفه بدهند.

سرویس خاتمه وقفه:

بعد از آنکه سرویس وقفه که معمولاً عمل انتقال است، کامل شد EDMA دو کار انجام میدهد:

۱- بیت مناسبی را در ثبّات CIPR، Set میکند.

۲- یک وقفه بنام EDMA_int به CPU میدهد. CPU با دریافت این وقفه روتین ISR مربوطه را اجراء میکند. البتّه اجرای این روتین مبتنی بر دو شرط است: set بودن بیت TCINTدر بخش OPT از PaRAM و set بودن بیت مناسب در ثبّات CIER. روتین ISR توسّط کاربر باید نوشته شود وگرنه CPU هیچ کاری انجام نمیدهد. در روتین ISR کاربر میتواند با بررسی CIPR متوجّه شود کدام وقفه کامل شده و بر اساس آن اقدامات مناسب را انجام دهد.

توضیح کامل EDMA را میتوانید در بیش از ۸۰ صفحه از [۳] مطالعه کنید.

۲-۱-۴) Timer:

DSP6711 دو timer بنامهای timer0 و timer1 دارد. کار timer این است که تا به تعداد مشخّص شده برایش بشمارد و وقتی بانتها رسید این موضوع را اعلام کند. خروجی timer میتواند به صورت پالسی یا clockای

باشد.timer برای شمارش از counter استفاده میکند. شکل ۲-۷ ،timer وساختار داخلی آنرا نشان میدهد. در این شکل :

۱-ثبّات TCR [1]:همان شمارنده timer است.

۲-Tinp:پایه ورودی timer است و میتوان از طریق آن مستقیما مقداری وارد counter کرد.

۳-Tout:پایه خروجی timer است و میتوان بدینوسیله مقدار Counter را دید.

۴-Tstat:خروجی است که با EDMA و CPU ارتباط دارد و میتواند آنها را دچار وقفه کند.

۵-HLD:میتوان بکمک این سیگنال از شمارش counter ممانعت نمود.

۶-GO: برای reset کردن counter

ثبّاتهای timer:

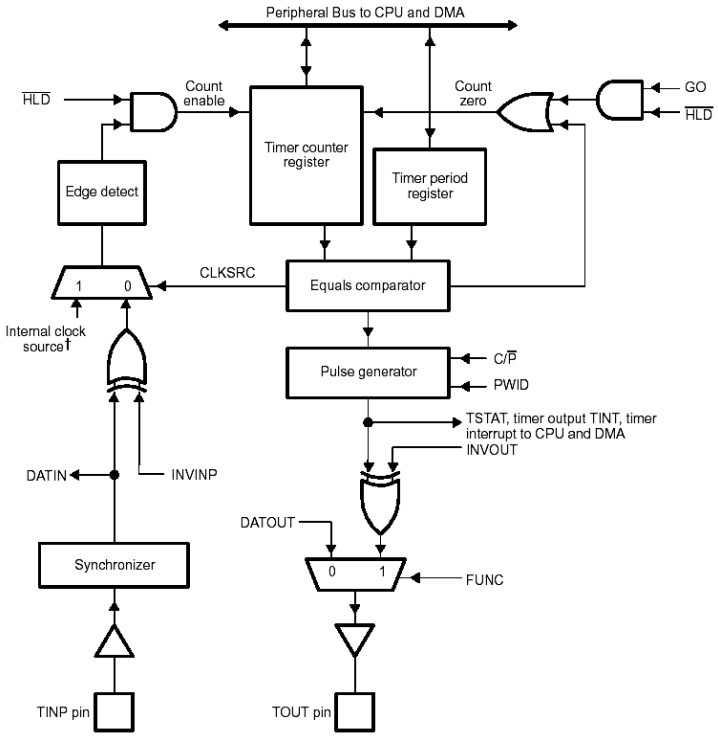
این ثبّاتها کار پیکربندی timer را انجام میدهند و عبارتند از:

۱- CTL [2]: برای کنترل timer بکار میرود. مواردی از قبیل فعّال کردن GO و Hold ، اینکه خروجی پالسی باشد یا به صورت clock ، تعریف چگونگی کار پایه های Tinp و Tout و .....

۲- PRD [3]: تعداد کلاکهایی که باید شمرده شود.

۳- CNT [4]:شماره فعلی counter را نشان میدهد.

[1] Timer Counter Register
[2] Timer Control register
[3] Timer Period register
[4] Timer Counter register


Peripheral Bus to CPU and DMA
HLD
Count enable
Timer counter register
Count zero
GO
HLD
Edge detect
Timer period register
1
0
CLKSRC
Equals comparator
Internal clock source†
Pulse generator
C/P
PWID
TSTAT, timer output TINT, timer interrupt to CPU and DMA
INVOUT
DATIN
INVINP
DATOUT
Synchronizer
0
1
FUNC
TINP pin
TOUT pin

شکل ۲-۷) timer

۲-۲) CCS 2[1]:

CCS یک IDE است که توسّط TI برای تسهیل کار با پردازنده های آن شرکت ، ارائه شده است. CCS شامل موادّ زیر است:

۱- کامپایلر ANSI C استاندارد

۲- اسمبلر برای تبدیل C به اسمبلی

۳- کتابخانه های سودمند برای هر چه بیشتر آسانتر کردن کار کاربر. مانند CSL، BSL، DSPLIB و... که توضیح مواردی از آنها خواهد آمد.

۴- محیط کنترل پروژه با امکانات عالی debug مانند امکان دیدن مقادیر متغیّرها از طریق watch window، امکان دیدن نمودارهای مختلف زمانی، فرکانسی و... توسّطexecution graph، امکان مشاهده میزان مصرف CPU توسّط CPU load graph، امکان بررسی تعداد دستورالعملها و زمان اجرای برنامه، تابع یا حتّی تکّه ای از برنامه و .....

۵- امکان تنظیم پارامترهای مربوط به EDMA، McBSP، EMIF و...بصورت گرافیکی DSP/Bios config که قادر است بر اساس این تنظیمات گرافیکی فایل c مناسب را بطور خودکار درست کند

۶- امکان استفاده از دستورات DSP/Bios که در real time شدن برنامه های ما کمک بسیار مؤثّری میکنند. بعنوان یک نمونه کوچک مثلاً اجرای دستور printf معمولی c بیش از ۱۰۰۰ سیکل clock پردازنده DSP6711 طول میکشد. امّا دستور Log_printf از دستورات DSP/Bios تنها حدود ۴۰ کلاک طول میکشد.

اسمبلر تعبیه شده در CCS قادر است بطرز بهینه ای کد c را به اسمبلی مورد استفاده توسّط پردازنده های DSP تبدیل کند. منظور از بهینه یعنی اینکه در این تبدیل قابلیّتهای این پردازنده ها مثل امکان اجرای موازی را مد نظر دارد.

جدیدترین نسخه CCS، CCS2 میباشد که در این پروژه از آن استفاده شده است. همچنین در این پروژه بنحو وسیعی از امکانات بالا استفاده شده است علی الخصوص موارد ۳ تا ۶.

[1] Code Composer Stodio

۲-۲-۱) کتابخانه CSL [1]:

مجموعه ای غنی از توابع و ماکروهاست که کار با ابزارهای داخلی پردازنده های DSP مثل EDMA، McBSP و... را بسیار آسان کرده است. بکمک این توابع دیگر لازم نیست ثبّاتهای این وسایل مستقیماً مقداردهی شود بلکه مثلاً با یک دستور mcbsp_init کلّی از این کارها انجام میشود.

توابع و ماکروهای CSL بصورت یک سری header file با نامهای مناسب در \ti\c6000\bios\include... قرار گرفته اند. مثلاً در CSL_EDMA.h توابع مربوط به پیکربندی و کار با EDMA آمده است. در این پروژه به وفور از این دستورات استفاده شده است. پاره ای از توابع CSL در جای خود توضیح داده خواهند شد. توضیحات کامل در [۶] موجود است.

۲-۲-۲)کتابخانه DSPLIB:

DSPLIB مجموعه است از توابع اساسی پردازش سیگنال که به صورت بهینه ای برای استفاده توسّط پردازنده های DSP نوشته شده است. نسخه مناسب آن برای DSP6711، C67xDSPLIB نام دارد. توابع پیاده سازی شده توسّط DSPLIB شامل مواردی از قبیل زیرند:

۱-فیلترهای وفقی

۲-correlation

۳-FFT ومعکوس آن

۴-پیاده سازی انواع فیلترها

۵-کانولوشن

۶-اعمال ریاضی: جذر، توان ۲، پیدا کردن ماکزیمم و....

۷-اعمال ماتریسی: ترانهاده، ضرب و ..

[1] Chip Support Library

و.....

Source های این کتابخانه در \ti\c6000\dsplib\src تحت نام dsp67x.src وجود دارد. اگر احیاناً در این پوشه DSPLIB مربوط به پردازنده دیگری مثلاً dsp62x.src بود میتوانیم این کتابخانه را بطور مستقل و بصورت یک فایل exe بنام c67xDSPLIB.exe از سایت شرکت TI [1] ، دریافت نماییم. بهمین وسیله همچنین میتوان همواره نسخه جدیدی از این توابع را در اختیار داشت.در این پروژه از جدیدترین نسخه آن که در فوریه ۲۰۰۳ منتشر شده استفاده شده است. توضیحات کامل مربوط به این کتابخانه در [۷] یافت میشود. در این پروژه از توابع FFT و IFFT این کتابخانه استفاده شده است.

۲-۲-۳) کتابخانه BSL [2]:

این کتابخانه مشتمل بر توابعی میباشد که امکان کار با اجزای DSK را فراهم میکنند.مثلاً مواردی از قبیل:

۱-توابعی برای برنامه ریزی سه LED که قبلاً ذکر شد.

۲-توابعی برای کار با حافظه flash مربوط به DSK

۳-توابعی برای کار با کدک صوتی

و.....

توضیح کامل این توابع در [۸] آمده است. از این توابع در پروژه ما استفاده نشده ولی چون موضوعی جالب توجّه بود برای تکمیل بحث آنرا ذکر کردیم.

[1] www.ti.com

[2] Board Support Library

۲-۳) نحوه ایجاد برنامه در ccs:

محیط ویرایش برنامه در CCS مشابهت بسیاری با IDE های ویژوال ویندوز مثل Visual C دارد.در اینجا برنامه ها در قالب یک پروژه قرار میگیرند. برای ساخت برنامه در این محیط مراحل زیر باید طی شود:

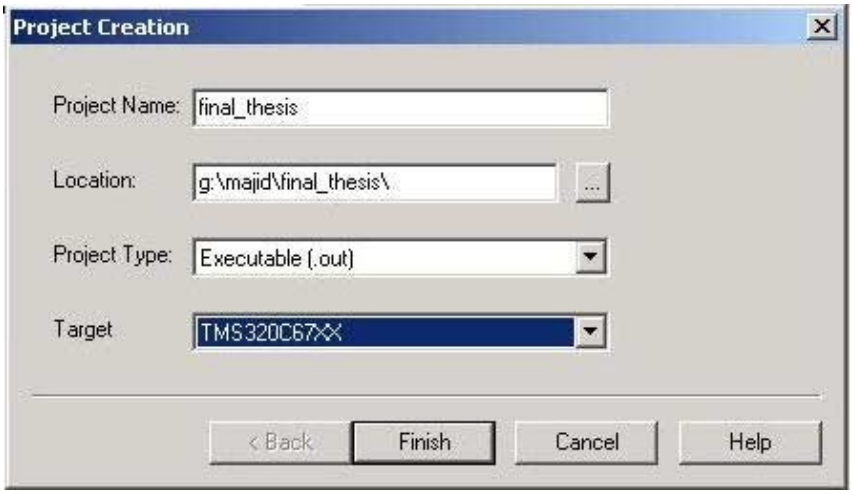
۱- با انتخاب منوی project/New صفحه زیر ظاهر میشود. فیلدهای این صفحه به صورت زیر باید پر شوند:

الف-Project Name: نامی دلخواه برای پروژه.

ب-Location:مکانی دلخواه جهت پروژه.

ج-File Type: باید Executable(.out) باشد.

د-Target: باید TMS320C67xx باشد.



شکل۲-۸) اوّلین قدم در ایجاد پروژه جدید

۲- توسّط منوی project\add file to projectفایلهای کتابخانه ای زیر به پروژه افزوده شوند:

..\ti\c6000\cgstools\lib\rts6711.lib

..\ti\c6000\bios\lib\csl_6711.lib

۳) کدهای C نوشته شده را از طریق همان منوی project\add file to project به پروژه اضافه کنید.

اکنون میتوانیم پروژه ساخته شده را با انتخاب project\Build ، build کنیم. نتیجه این کار یک فایل اجرایی CCS با پسوند out. میباشد. قبل از build کردن در project\Build option باید تنظیمات زیر برقرار باشند:

۱- در compiler>Basic :

Target Version:671x

۲- در compiler>Processor:

Define Symbols:CHIP_6711

سایر پارامترها معمولاً مقدار مناسب خود را دارند. پس از انجام این تنظیمات پروژه را build میکنیم و در صورت نبود خطا فایل out. درست خواهد شد.

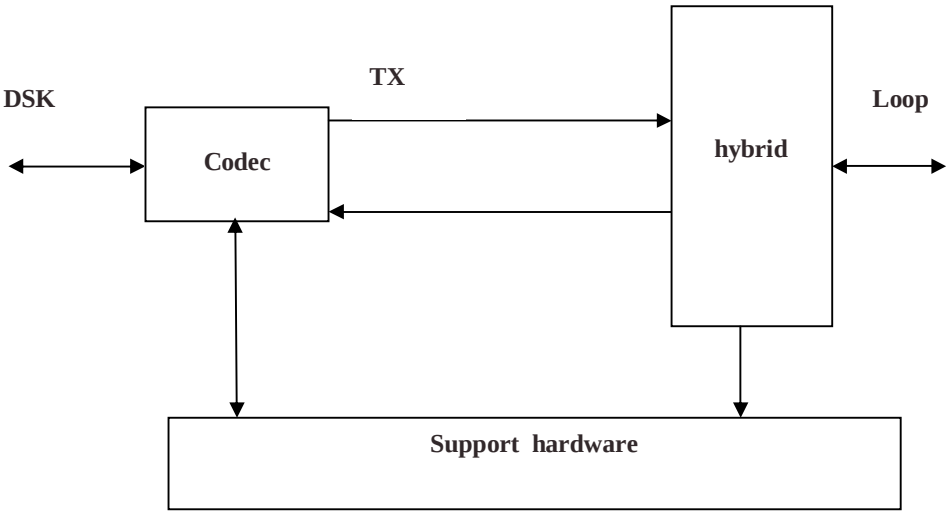
۳)طرّاحی سخت افزاری:

در این بخش نحوه طرّاحی بخش آنالوگ مدار را که اصطلاحاً AFE [1]، بررسی خواهد شد. این بخش واسط میان DSK و خط تلفن میباشد. کار خود را با یک بلوک دیاگرام کلّی شروع کرده و در نهایت به طرّاحی کامل آن خواهیم رسید.

۳-۱) بلوک دیاگرام کلّی سیستم:

شکل ۳-۱ ساختار کلّی AFE را نشان میدهد:

[1] Analog Front End



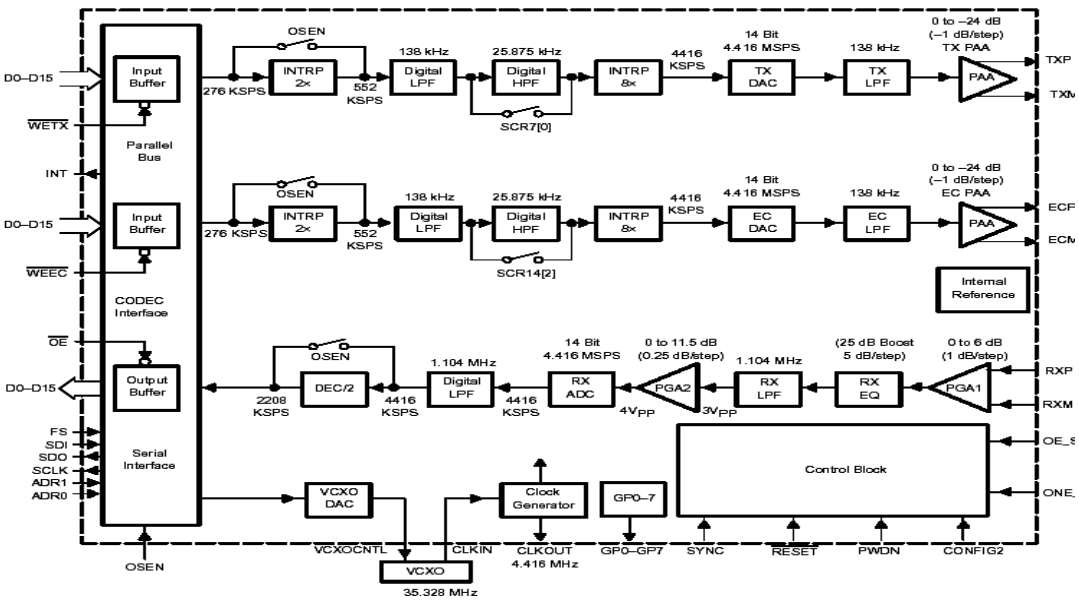
شکل ۳-۱)بلوک دیاگرام کلّی AFE

۳-۲) توضیح اجزاء:

۱-codec : اصلیترین بخش AFE است. وظیفه این واحد این است که داده های دیجیتالی DSK را به سیگنالهای آنالوگ مناسب برای خطّ تلفن تبدیل کند و بالعکس سیگنالهای خطّ تلفن را به داده های دیجیتال. در نتیجه این کدک باید شامل D/A و A/D باشد. همچنین در کدک ما باید یک سری فیلتر برای هدایت سیگنال به پهنای باندهای مناسب برای ارسال و دریافت در ADSL، داشته باشد.

خوشبختانه شرکت TI با ارائه IC کدک مودم ADSL همه این مشکلات را حل کرده است. این IC با نام TLV320AD11 میتواند مستقیماً با DSK مربوط شود، توسّط آن برنامه ریزی شود و تمام کارهای گفته شده در بالا بطور خودکار انجام دهد.

شکل ۳-۲ بلوک دیاگرام این کدک را نشان میدهد. همچنین در انتهای این بخش نحوه اتّصال آن به DSK را با ذکر شماره پایه های آن بصورت شماتیک ORCAD رسم کرده ایم.



شکل ۳-۲) کدک TLV320AD11

مهمترین اجزای کدک و موارد مرتبط با آنها:

۱الف-دو کانال گیرنده داده از DSK : در یکی از این دو کانال امکانات حذف اکو فراهم شده است.

این دو میتوانند بسته به نوع تنظیم داده را با سرعتهای ۲۷۶ یا ۵۵۲ کیلو بیت برثانیه از DSK دریافت کنند یک فیلتر low pass با پهنای باند ۱۳۸ KHZ برای جداسازی پهنای باندDownstream بکار رفته است. همچنین یک فیلتر High pass برای جداسازی باند صوت میتواند بکار رود.بخش اصلی این کانالها هم که واضحا A/D آن میباشد. همچنین چند تقویت کننده دارد که درصورت لزوم میتوان از آنها استفاده نمود.

ب-یک کانال فرستنده داده به کدک: که شامل یک فیلتر low pass برای جداسازی upstream ، D/A و چند تقویت کننده است که در شکل با ذکر مقادیر لازمه مشخّص میباشند.

ج-واسط سری: برنامه ریزی کدک با استفاده از چند ثبّات ۱۶ بیتی تعبیه شده در آن صورت میگیرد. ریختن مقدار در این ثبّاتها بصورت سریال انجام میگیرد. McBSP میتواند

بهترین گزینه برای اینکار باشد. بدین منظور سیگنالهای Frame Synch و CLK هم در نظر گرفته شده که این امکان را فراهم میکند که بدون هیچ مشکلی مستقیماً از طریق McBSP این برنامه ریزی انجام شود.

د-واسط موازی: برای انتقال داده استفاده میشود و چون در DSK این کار بکمک EMIF صورت میگیرد امکانات مناسب از قبیل سیگنالهای REDDY، RE وWE در آن تعبیه شده.

ه- clock کدک بصورت خارجی باید تأمین شود. این clock باید یک VCXO با فرکانس 35.328 MHZ ± 50 ppm و minimum duty cycle برابر ۶۰/۴۰ باشد. CSX750 IC ساخت شرکت سیتیزن یک IC مناسب بدین منظور میباشد.

و-DAC کنترل کننده VCXO: از طریق این D/A میتوانیم فرکانس clock خارجی را ثابت نگهداریم و از تغییرات آن جلوگیری کنیم.بکمک ثبّاتهای SCR4 و SCR5 کدک data مناسب اینDAC، فراهم میشود.

ز-ثبّاتها: که همانطور که گفته شد برای برنامه ریزی کدک بکار میروند. از این ثبّاتها که SCR0 تا SCR14 نام دارند موارد زیر در پروژه ما استفاده شده است:

SCR1: تنظیم بهره TX PAA

SCR2: تنظیم بهره RX PGA2

SCR4 و SCR5: تنظیم rate کلاک

SCR9 و SCR10: تنظیم جبران [1] کانال RX

SCR11: انتخاب بهره کانال TX

SCR12: بهره RX PGA1

[1] Offset

در بخش طرّاحی نرم افزار مقادیر مناسب این ثبّاتها را نشان میدهیم. اطّلاعات کامل در مورد این کدک در [۹] موجود است. این کدک برای قرار گرفتن در تأسیسات مشتری طرّاحی شده است. مدل با طرز عملکردی مشابه آن بنام TLV320AD12 برای قرارگرفتن در مرکز تلفن طرّاحی شده است.

نکته مهم دیگر این که دامنه خروجی TX برای ارسال روی خطّ تلفن کوچک است و باید بکمک Line driver آنرا تقویت کنیم. باز هم خوشبختانه TI ، IC بنام THS6022 را عرضه کرده که نحوه استفاده از آن با ذکر شماره پایه هایش در انتهای این بخش بصورت شماتیک ORCAD نشان داده شده است و اطّلاعات کامل در مورد آن را میتوانید در [۱۰] پیدا کنید.

۲- Hybrid: برای تبدیل ۴ سیم به دو سیم بکار میرود تا بتوانیم ارسال و دریافت را رو ی یک زوج سیم معمولی تلفن انجام دهیم. طرّاحی یک مدار کامل hybrid که برای حالات مختلف کارائی داشته باشد کار مشکلی است. یک مدار ساده که در پروژه هم از همان استفاده شده در [۲۶] آمده است. در این مدار فرض شده است تمام پهنای باند را در اختیار داریم و تکنولوژی دیگری مثلاً VDSL ، از کانال استفاده نمیکند. در همین مرجع چند مدار پیچیده تر برای سایر حالات توضیح داده شده است.

۳- Support Hradware: این بخش ابزارهای جانبی کار را شامل میشود:

الف-clock: توضیح آن گذشت.

ب-منابع تغذیه وسایل: خوشبختانه منابع تغذیه لازم را میتوان از طریق DSK تأمین کرد. AFE ما به ولتاژهای تغذیه ۵ ، ۱۲، ۱۲- و ۳,۳ ولت نیاز دارد که DSK همه اینها را از طریق واحد expansion peripheral در اختیار ما گذاشته است. برای اینکه کدک بهترین کارائی خود را داشته باشد لازم است پایه های تغذیه ۳,۳ ولتی آن از طریق ۴ منبع مختلف تأمین شوند. میتوان در برد بکمک Ferrite bead ها از یک منبع ۳,۳ ولتی، چهار انشعاب ۳,۳ ولتی گرفت این امر در شماتیک مداری که در پایان این بخش رسم گردیده آمده است.

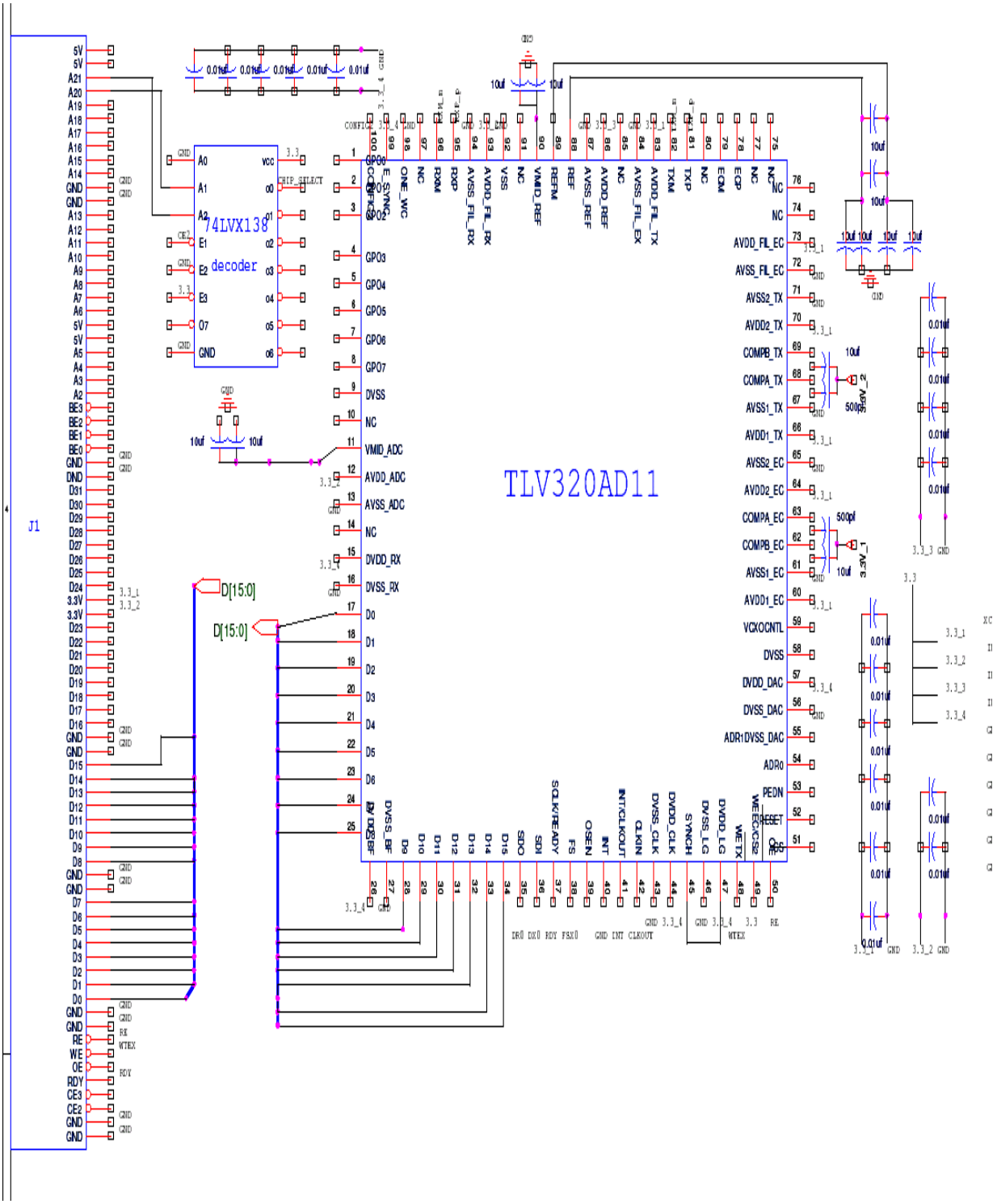
ج-یک فلیپ فلاپ برای تأخیر دادن پایه INT کدک که به XINT4 وصل شده است تا دادن وقفه به EDMA در زمان مناسب انجام گیرد. برای اطّلاعات بیشتر به [۹] مراجعه شود. IC مورد استفاده باید ولتاژ پایین [1]باشد تا بتوانیم از تغذیه ۳,۳ ولتی که داریم استفاده کنیم. یک مورد مناسب 74LV574D است که در این پروژه از آن استفاده شده است.

د-یک معکوس کننده برای استفاده در موارد مورد نیاز. این معکوس کننده هم باید ولتاژ پایین باشد. ما از 74LVT14 استفاده کرده ایم.

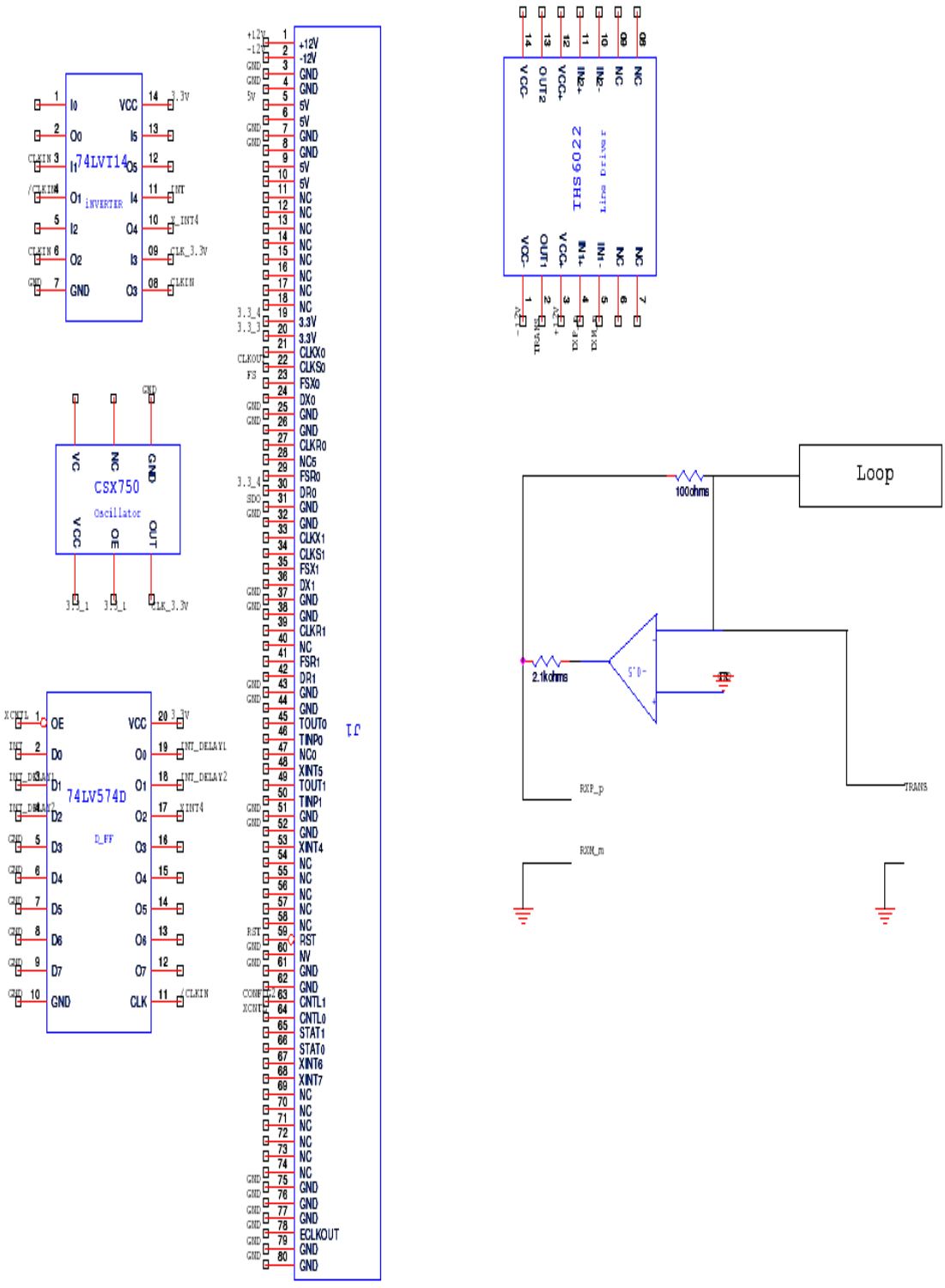
ه-دیکدر ولتاژ پایین مثل 74LVX138 برای انتخاب IC ها در موارد مناسب.

در انتهای این بخش شماتیک کامل AFE را که با ORCAD ترسیم کرده ایم، می آوریم.

[1] low voltage



شکل ۳-۳) شماتیک ORCAD مربوط به AFE



شکل ۳-۳) (ادامه) شماتیک ORCAD مربوط به AFE

۴)توضیح نرم افرار:

برنامه ای که توضیح آن خواهد آمد به زبان C در محیط CCS2 نوشته شده است. سعی شده با حدّاکثر بهره گیری از بخشهای مختلف پردازنده DSP و امکانات بسیار مفید CCS، برنامه ای با قابلیّت بالا نوشته شود. این بخش به توضیح نرم افزار نوشته شده اختصاص دارد.

۴-۱) الگوریتم کلّی:

۱- اقدامات قبل از شروع کار ارسال و دریافت:

الف-پیکربندی McBSP0 برای اینکه بتوانیم با ارسال مقادیر توسّط آن، کدک را پیکربندی کنیم.

ب-پیکر بندی EMIF، برای اینکه امکان ارتباط با کدک و سایر IC های AFE فراهم شود.

ج-پیکربندی کانالهای وقفه خارجی EXT5 و وقفه Timer1 از EDMA برای اینکه بتواند ارسال و دریافت داده را انجام دهد.

د-پیکربندی Timer1 برای زمانبندی ارسال

ه-پیکربندی کدک از طریق مقداردهی ثبّاتهای SCR0 تا SCR14

و-پیکربندی وقفهEDMA_ INT مربوط به CPU .

۲-برنامه بخش ارسال:

الف-یک فریم داده را از فایل بردار و آن را در یک آرایه بریز.

ب-اعمال مدولاسیون DMT

۱-ب-اعمال bit allocation

۲-ب-اعمال constellation encoding

۳-ب-انجام IFFT

ج-Timer0 را فعّال کن تا شروع بشمارش کند.

د-هنگامیکه Timer0 به مقدار مورد نظر رسید وقفه EXT4 به طور خودکار کار خود را شروع میکند و داده را از آرایه گفته شده در بالا به بافر ورودی کدک ریخته میشود.

۳-برنامه بخش دریافت:

الف-با رسیدن داده کدک پایه وقفه EXT5 از EDMA را فعّال میکند.

ب-با فعّال شدن این وقفه مقادیر از کدک دریافت شده و در مکان حافظه ای که در پیکربندی کانال EXT5 مشخّص شده ریخته میشوند.

ج-عکس مدولاسون DMT:

۱-ج- FFT

۲-ج-constellation decoding

۳-ج-عکس bit allocation .بعد از این عمل فریمهای مربوطه بدست می آید.

این مراحل کلّی انجام برنامه بود. در ادامه بحث هرکدام از آنها بتفصیل شرح داده میشوند.

۴-۲) پیکربندی McBSP:

برای پیکربندی McBSP و دیگر بخشهای پردازنده که توضیح آنها خواهد آمد از محیط گرافیکی که CCS2 با ارائه فایلهای پیکربندی DSP/BIOS برای ما فراهم کرده استفاده کرده ایم. با این تمهید دیگر لازم نیست بصورت صریح مثلاً با استفاده از دستورات انتقال ثبّات این کار را انجام دهیم. بلکه این کار را در یک محیط گرافیکی انجام میدهیم. CCS خود بطور خودکار فایل C مناسب را تولید میکند.

در این محیط گرافیکی ابتدا یک شیء McBSP configuration ایجاد میکنیم. خواصّ این شیء همان مقادیر پیکر بندی McBSP است.

در اینجا میخواهیم پیکربندی McBSP را بصورتی انجام دهیم که بتوانیم به بخش سریال کدک داده منتقل کنیم. بدین منظور موارد زیر باید انجام میشد.

الف-از طریق expansion peripheral پایه DX از McBSP به پایه SDI از کدک وصل میشود.

ب-باانتخاب گزینه Internal clock در شیء پیکربندی McBSP، میگوییم clock ارسال داده توسّط خود McBSP تولید شود.

ج-Frame synch mode را sample rate generator انتخاب میکنیم تا Frame synch توسّط این واحدتولید شود.

د-FSGM را در DXR to XSR copy تنظیم میکنیم. چون میخواهیم Frame synch توسّط sample rate generator وقتی تولید شود که DXR در XSR کپی میشود.

ه-سایز عناصر را ۱۶ بیت تنظیم میکنیم. چون ثبّاتهای کدک ۱۶ بیتی میباشند.

و-میخواهیم انتقال تک فاز معمولی داشته باشیم. در نتیجه در شیء، گزینه single phase را انتخاب میکنیم.

ز-گزینه Digital loop back باید disable باشد. چون در این حالت DR و DX از McBSP به یکدیگرمتّصل میشوند.

برای اطّلاعات بیشتر به توضیحات مربوط به McBSP در بخشهای گذشته مراجعه شود.

۴-۳) پیکربندی Timer1:

در این پروژه از Timer1 برای زمانبندی ارسال استفاده میشود. چون بر خلاف فاز دریافت که بصورت آسنکرون انجام میشود و هر زمان که کدک، داده آماده داشت انجام میشود، دریافت دست خودمان است و بهمین خاطر Timer1 را طوری تنظیم کرده ایم که تا حدّ مناسب بشمارد. پیکربندی Timer1 را طوری انجام میدهیم که هر وقت به زمان مورد نظر رسید وقفه Timer1 از EDMA را فعّال کند. احضار این وقفه همانطور که بزودی گفته میشود میتواند موجب ارسال داده به کدک شود.

۴-۴) پیکربندی EDMA:

در این پروژه از دو کانال وقفه EDMA استفاده میشود. یکی از آنها کانال XINT5 است که با فعّال شدن این وقفه توسّط کدک،EDMA داده ها را دریافت میکند. وقفه دیگر وقفهTimer1 است. وقتی Timer1 که در مورد قبل توضیح دادیم به زمان خود رسید وقفه Timer1 از EDMA فعّال میشود. بکمک امکان LINK وقفه های EDMA ،این وقفه را طوری تنظیم میکنیم که با Link به آدرسی دیگر در PaRAM موجب ارسال داده میشود.

مقداردهی PaRAM در این دو وقفه را هم با استفاده از فایل پیکربندی BIOS بصورت گرافیکی بصورت زیر انجام دادیم. برای وقفه XINT5:

۱- TINT:enable

۲- TCC=11

۳- FC(Frame count)=0

۴- Element count=320 H

۵- 2Ds=0

۶- Element Size(ESize)=16 bit

۷- SUM(Source Address Mode):none

۸- DUM(Destinatio Address Mode):Increment

۹- Source Address: A0000000

۱۰- Destination: آرایه ای بنام data_rec

در مورد ۱و۲ وقفه تکمیل دریافت ۱۱ را فعّال میکنیم. در مورد ۳و۴و۵ میگوییم هدف ما انتقال یک بعدی با تعداد فریم برابر ۱ و تعداد عناصر ۳۲۰H است. در مورد ۶ میگوییم میخواهیم عناصر ۱۶ بیتی انتقال دهیم. چون همانطور که میدانیم گذرگاه داده در کدک، ۱۶ بیتی است. در مورد ۷ با انتخاب گزینه None میگوییم آدرس مبدأ برای ارسال بعدی ثابت است. چون مبدأ ما کدک است و همواره باید از بافر آن بخوانیم. امّا درمورد بعد آدرس بعدی برای انتقال با اضافه کردن به آدرس فعلی انتقال بدست می آید. چون مقصد ما حافظه DSK است. با توجّه به مورد ۶ یعنی هر بار ۲ واحد (دو بایت=۱۶ بیت) به آدرس اضافه میشود.

در مورد ۹ میگوییم مبدأ ما آدرس A0000000H است. این آدرس فی الواقع کدک را مشخّص میکند. چون قبلاً گفتیم برای آدرس دهی کدک از فضای حافظه ای CE2 استفاده میکنیم. این محدوده حافظه ای آدرسهای A0000000H تا A3FFFFFF را دربرمیگیرد. از طرفی همانطور که در شماتیک ORCAD در بخشهای قبل نشان داده شده است، از پایه های آدرس A20 و A21 برای فعّال سازی کدک استفاده میشود . وقتی که این پایه ها low باشند و استفاده از فضای CE2 توسّط EMIF بنحوی که گفته میشود فعّال شده باشد، کدک انتخاب

میشود. در نتیجه تمام محدوده های فضای CE2 که در آنها A20=A21=0 باشد میتوانند کدک را فعّال کنند که ما از آدرس A0000000H استفاده کرده ایم.

در مورد ۱۰ میگوییم مقصد ما آرایه ای در حافظه بنام data_rec است. CCS به ما امکان میدهد که آدرس را هم بطور صریح و هم مثل اینجا بصورت سمبلیک تعریف کنیم. حسن مهم تعریف سمبلیک این است که با مشکل بدست آوردن فضای مناسب آدرسی روبرو نمیشویم. ما مثل تعریف آرایه در C ، یک آرایه تعریف میکنیم و نام آنرا که همانطور که میدانیم به اوّلین عنصر آرایه اشاره میکند به عنوان آدرس سمبلیک در پیکربندی وارد میکنیم .

برای پیکربندی وقفه Timer1 مربوط به EDMA کار خاصّی لازم نیست چون نمیخواهیم با خود آن انتقال را انجام دهیم.

همچنین در این پروژه از توابع مختلف کتابخانه CSL استفاده شده است. مثلاً به وسیله تابع EDMA_Link به وقفه Timer1 میگوییم هنگام رخدادن وقفه Timer1 به آدرسی در حافظه PaRAM برو و آن وقفه را سرویس بده. در آن وقفه پیکربندیهای لازم برای عمل send قرار داده میشود. یا مثلاً تابع EDMA_intDisable با غیر فعّال کردن بیت مناسب در CIER، وقفه خاتمه انتقال را غیر فعّال میکند. یا تابع EDMA_intClear که بیت مورد نظر در CIPR را غیر فعّال میکند.

مقادیر PaRAM به صورت زیر میباشند:

TINT:enable
TCC=10
FC(Frame count)=0
Element count=82H
2Ds=0
Element Size(ESize)=16 bit
SUM:Increment

DUM:None

Source Address: آرایه ای بنام send_data

Destination: A0000000

برای توضیحات بیشتر در مورد EDMA میتوانیدبه بخش آشنایی با پردازنده ... در همین پایان نامه و مراجع دیگر معرّفی شده در آنجا مراجعه نمایید.

۴-۵) پیکربندی EMIF:

پیکر بندی EMIF ، از EDMA آسانتر است. کافیست در فایل پیکربندی DSP/BIOS موارد زیر را انجام دهیم.

۱- در گزینه CE2 space ، .Memory type=16 bit asynch انتخاب میکنیم. چون کدک ما که میخواهد از فضای حافظه ای CE2 استفاده کند یک وسیله آسنکرون است و از نوع حافظه نیست.

۲- CLKOUT:Held High تا غیر فعّال باشد.

۳- CLKOUT=Enable to clock

بقیّه فیلدها در مقدار پیش فرض میمانند.

۴-۶) پیکربندی کدک:

همانطور که گفته شد برای پیکربندی کدک باید ثبّاتهای SCR آن پیکربندی شوند. برنامه ریزی ثبّاتها توسّط McBSP از طریق واسط سری آن انجام میگیرد. ثبّاتهای مورد استفاده و مقادیرمناسبشان در زیر آورده شده است:

۱- SCR0=0000H

۲- SCR2=0402H

۳- SCR4=087FH

۴- SCR5=0A7FH

۵- SCR6=0CFFH

۶- SCR7=0F00H

۷- SCR8=1400H

۸- SCR10=1200H

۹- SCR14=1C06

تابع mcbsp_write از کتابخانه CSL برای ریختن مقادیر در ثبّاتها استفاده شده است. ثبّاتهای کدک ۸ بیتی میباشند.البتّه مقادیر ارسالی به ثبّاتها ۱۶ بیتی میباشند. ۸ بیت اوّل برای مشخّص کردن اینکه کدام ثبّات انتخاب شده است و همچنین برای مشخّص کردن این است که میخواهیم در ثبّات بنویسیم یا از آن بخوانیم .در مورد توضیح بیشتر این مواردو همچنین تفسیر مقادیر ثبّاتها به بخش آشنایی با پردازنده ... و مراجع اشاره شده در آنجا مراجعه کنید.

همانطور که در بخش آشنایی با پردازنده اشاره شد، وقتی به وقفه های EDMA ، سرویس مناسب داده شد، وقفه EDMA_INT به CPU فرستاده میشود و در آنجا روتین ISR مربوطه اجرا خواهد شد. امّا همانطور که خواندیم نوشتن ISR به عهده خودمان است. حتّی CPU، مقدار set شده در CIPR را هم پاک نمیکند و همه کارها توسّط خودمان انجام میگیرد.

در روتین ISR ابتدا با استفاده از تابع EDMA_intTest از توابع CSL چک میکنیم که کدام وقفه فرا رسیده است. اگر وقفه اتمام ارسال رسیده بود با استفاده از تابع send یک بخش داده دیگر را برای ارسال آماده میکنیم. اگر وقفه اتمام دریافت رسیده بود، کار خاصّی لازم نیست انجام شود. برای وقفه Timer1 هم چون در اینجا مهم نیست، اصلاً فیلد CIER اش را غیر فعّال میکنیم که وقفه اتمام ندهد.

۴-۷) پیاده سازی DMT:

در بخشهای گذشته موارد سخت افزاری مورد استفاده بررسی شد. در اینجا به مسئله پیاده سازی مدولاسیون DMT میپردازیم. در این رابطه طرز کار توابع نوشته شده را توضیح میدهیم.

اطّلاعاتی که باید ارسال شوند را در یک فایل بنام test.txt قرار داده ایم.فریمی که باید ارسال شوند ابتدا ازاین فایل خوانده میشوند و در آرایه input_data قرار میگیرند. تابع ()send که در فایل send.c قرار گرفته وظیفه ارسال داده به کدک را برعهده دارد. طول فریم ارسالی ۱۳۲ بیت است. وظیفه تخصیص بیت توسّط تابع words2bits که در فایل words2bits قرار دارد انجام میشود. تعداد بیتهای تخصیصی به هر یک از ۳۲ کانال مورد استفاده در اینجا را در یک فایل بنام send_chanspec.dat قرار داده ایم که در ابتدای برنامه با احضار تابع initialize_channel در آرایه bits_per_snd_channel قرار میگیرد. در words2bit بر اساس مقادیر این آرایه، بیتها از input_data خوانده میشوند. به ازای هر کانال یک مقدار short int برگردانده میشود که نشاندهنده بیتهای تخصیص یافته به آن کانال است. مثلاً اگر قرار است ۴ بیت به کانالی تخصیص یافته شود و آن چهار بیت 0010 باشد. عدد ۲ به این کانال داده میشود. بدینترتیب یک آرایه بنام channel بوجود می آید.

تا اینجای کار تخصیص بیت انجام شد. مرحله بعد انجام constellation encoding است. در این برنامه ابتدا برای Constellation ها جداولی در ابتدای برنامه درست میشود. همانطور که قبلاً اشاره شد برای درست کردن constellation سه حالت b زوج، b=3 و b فرد بزرگتر از ۳ باید در نظر گرفته شود. در اینجا هم همین امر پیاده شده است. در آرایه دو بعدی بنام maps، constellation ها ساخته میشوند. آرایه های maps[2]، maps[3] و maps[5] صریحاً مقدار دهی میشوند. آنگاه constellation های زوج بر اساس دو بیتی و constellation های با تعداد فرد بر اساس ۵ بیتی بدست آورده میشوند. این هر دو کار توسّط تابع generate_constellation که در فایل maps.c قرار دارد، انجام میشود. در مرحله بعد توابع

generate_even_lookup_table و generate_odd_lookup_table جداولی تولید میکنند که بر اساس آنها میتوان کد QAM تخصیص یافته را بدست آوریم.

تابع encode_list بر اساس این جداول کد QAM مناسب را تخصیص میدهد. خروجی این واحد ۳۲ مقدار مختلط است با مؤلّفه های از نوع short int. پیاده سازی نوع complex در فایل header بنام complex.h براحتی انجام شده است. بعد از کد کردن داده ها، از این داده ها بهمراه مزدوجشان IFFT گرفته میشود. برای IFFT از تابع icfftr2_dif ، موجود در جدیدترین نسخه توابع DSP_LIB که توضیح آنها گذشت استفاده شده است. این توابع و توابع مربوطه در فایل ff2.c قرار گرفته است. اگر به این توابع مراجعه کنیم می بینیم ورودی آنها اعداد حقیقی از نوع float میباشد ولی داده های ما اعداد مختلط با مؤلّفه های short میباشند. اگر کمی دقّت کنیم میبینیم این مسئله اصلاً مشکلی ایجاد نمیکند. چون دو مقدار short روی هم ۴ بایت میشود و یک مقدار float هم ۴ بایت است. و چون ورودی این توابع هم اشاره گر به float است اصلاً مشکلی پیش نخواهد آمد. بعد از انجام IFFT ، داده قابل ارسال بدست آمده است.

در بخش دریافت فی الواقع معکوس این کارها صورت خواهد گرفت. تابع bit2words ما را به فریم اصلی میرساند. تابع decode_list ، constellation decoding را انجام میدهد و بالاخره کار FFT هم که واضح است.

۴-۸)بررسی نتایج:

۱- همانطور که قبلاً گفته شد تعداد بیتهای تخصیصی در واحد تخصیص بیت تا ۱۵ بیت میرسد. این مستلزم تعریف تعداد نقاط constellation زیاد میباشد. در برنامه نوشته شده هنگام تعریف آرایه های مربوط به constellation های بیش از ۷ بیت، هنگام Link ، برنامه دچار خطایی به صورت زیر میشد:

"realloctio value truncate in….." با حذف این آرایه ها، خطاها رفع میشدند. بهمین علّت تخصیص بیت بیش از ۷ بیت مقدور نشد.

۵) خلاصه،نتیجه گیری و پیشنهادات

همانگونه که در ابتدا اشاره شد، این پایان نامه فی الواقع بخش دوّم پروژه "طرّاحی و پیاده سازی مودم ADSL میباشد" که بخش اوّل آن با عنوان "تخمین طیف کانال در مودم ADSL" در پایان نامه ای مجزّا اتجام شده است. در این پایان نامه طرّاحی نرم افزار و سخت افزارهای لازم انجام شده است.

درابتدا کار حود را با ذکر ساختار کلّی مودم آغاز کردیم. ساختار کلّی آن را بیان کردیم و اجزای مختلف را توضیح دادیم. بخش بعدی را به مروری برپردازنده TMS3206711 و DSK TMS3206711 که یک starter kit برای این پردازنده است، اختصاص دادیم. در این پروژه واحد DMT که قلب مودم ADSL بشمار میرود، بر روی این پردازنده پیاده شده است.استفاده از پردازنده های DSP در این بخش که با واخدهایی مانند IFFT و FFT سروکار دارد، امری ضروری بشمار می آید تا امکان کار کردن real time این واحد فراهم آید. بعلاوه با تمهیداتی که در بورد پشتیبان پردازنده–DSK– فراهم گردیده است، میتوان به سهولت daughter board پروژه را به آن متّصل کرد.

برای داشتن یک برنامه کارا، سعی شده از قابلیّتهای DSP و DSK تا آنجا که مقدور بوده استفاده شود. از جمله:

۱–استفاده از واحد ارتباط سریال McBSP برای برنامه ریزی کدک مورد استفاده.

۲-استفاده از یکی از دو timer موجوددر DSP6711 برای زمانبندی ارسال. چون ارسال را میتوان بصورت منظّم انجام داد و فقط دریافت است که بصورت آسنکرون انجام میشود.

۳-استفاده از واحد DMA بهبود یافته(EDMA) تا امکان ارسال و دریافت بدون نیاز به CPU فراهم گردد. با تعریف پیکربندیهای مناسب برای کانالهای EDMA، عمل ارسال و دریافت انجام میشود.

۴-استفاده از EMIF: که امکان ارتباط DSP با حافظه و ابزارهای memory map را فراهم میکند. این کار برای اینکه بتوانیم با برد جانبی ارتباط برقرار کنیم لازم می آید.

نرم افزار مزبور در محیط CCS2 پیاده شده است. CCS2 با داشتن یک کامپایلر ANSI C ، به ما امکان میدهد برنامه های خود را به زبان C بنویسیم. CCS2 سپس آنرا به اسمبلر پردازنده تبدیل میکند. بعلاوه آن چیزی که از همه مهمتر است این است که CCS2 با داشتن کتابخانه ای غنی، برنامه ریزی و پیکربندی واحدهای مختلف پردازنده را بسیار آسان کرده است. کتابخانه هایی مانند CSL، BSL و DSPLIB که در این پروژه از مورد اوّل بکرّات و از سوّمی برای پیاده سازی FFT و IFFT استفاده شده است.

همچنین در این پروژه از امکاناتی دیگر مانند استفاده از فایلهای پیکربندی DSP/BIOS، Profiler و ....نیز استفاده شده است.

در بخش سوّم این پروژه با طرّاحی برد جانبی آشنا میشویم. این برد به ما امکان میدهد که خروجی DSK را به خطّ تلفن انتقال دهیم. کار را با ارائه ساختار کلّی آغاز کردیم و در انتها به طرح کاملاً واقعی و کامل رسیدیم که شماتیک آن که بکمک نرم افزار ORCAD ترسیم گردیده است، در انتهای همان بخش آمده است.

اجزای این بخش عبارتند از:

۱-کدک: شامل A/D و D/A و فیلترهای لازم برای جداسازی upstream، downstream. جهت انجام این امور از TLV320AD11 که توسّط شرکت Texas استفاده شده است. کدک بخش اصلی این واحد بشمار میرود.

۲-Line Driver: برای تقویت خروجی کدک تا سیگنال ورودی خطّ تلفن بحدّ مناسبی برسد.

۳-Hybrid: برای تبدیل ۲ سیم به ۴ سیم.

بالاخره در بخش چهارم نرم افزار طرح توضیح داده شده است. نحوه پیکربندی هر یک از اجزای مورد نیاز پردازنده در این فصل تشریح شده است. همچنین توابع پیاده کننده اجزای DMT ، مانند واحد bit allocation، constellation encoding,decoding و..... پیاده شده است. توضیحات بیشتر در مورد نحوه تست و سرعت بدست آمده و مشکلات طرح در بخش ۴ آمده است.

با این ترتیب میتوان گفت در این پروژه طرّاحی کامل یک مودم ADSL قرارگیرنده در تأسیسات مشتری انجام گرفته است.

همچنین میتوان بعنوان ادامه کار:

۱- پیاده سازی مودم ADSL قرار گیرنده در مرکز تلفن مورد بررسی و پیاده سازی قرار داد. شرکت TI، کدک مورد استفاده در مرکز تلفن را هم با نام TLV320AD12 تولید کرده است.

۲- نوشتن درایور مربوطه تا بتوانیم بهمراه مورد ۱، یک طرح کاملاً واقعی داشته باشیم.

۳-بدست آوردن الگوریتمی برای اینکه بتوانیم Constellation encoding را طوری پیاده کنیم که حافظه کمتر مصرف کند .

مراجع:

[1]" Network and Customer Installation Interfaces –Asymmetric Digital Subscriber Line (ADSL) Metallic Interface" , ANSI T1.413 -1998

[2]"ASYMMETRICAL DIGITAL SUBSCRIBER LINE (ADSL) TRANSCEIVERS" , ITU G.992.1:, July 1999

[3] "TMS320C6000 Peripherals Reference Guide", Texas Instruments, Literature Number: SPRU190D
Feb. 2001

[4] "TMS320C6000 Peripherals Reference Guide Manual Update Sheet", Texas Instruments, Literature Number SPRZ122C Jan. 2003

[5] "TMS320C6000 McBSP Initialization", Texas Instruments, Application Report Number SPRA488B - April 2002

[6]" MS320C6000 Chip Support Library API User's Guide", Texas Instruments, Literature Number SPRU401B- April 2001

[7]"TMS320C67x DSP Library Programmer's Reference Guide", Texas Instruments, Literature Number SPRU657- February 2003

[8] "TMS320C6000 DSK Board Support Library API User's Guide", Texas Instruments, Literature Number SPRU432A - October 2001
[9] "TLV320AD11A 3.3 V INTEGRATED ADSL OVER POTS CODEC" , Texas Instruments, Number SLWS087B March 2000

[10]"THS6022 250-mA DUAL DIFFERENTIAL LINE DRIVER," , Texas Instruments, Number SLOS225C , JANUARY 2000

[11]"ADSL Full Rate and D.Lite General Office Modem Analog Front End Using TLV320AD12", Texas Instruments, Number SLWA015, August 1999

Abstract:

This project is the second section of a project called “Design and implementation of an ADSL modem”. The first one with the name of “Telephone channel modeling in an ADSL modem” carried out in a separate thesis.

This project discuss about the hardware and software sections of ADSL modems. In hardware section we design almost all the hardware sections needed. We use a DSP processor called TMS3206711 and a daughter board beside it. In software section we implement the needed program by using an environment which is provide by Texas instrument to help DSP processors programming. This is called CCS2.

Sharif University of Technology

Faculty of Computer Engineering

M.S.C. Thesis

Design and Implementation of an ADSL Modem

Abolfazl Shams

Supervisor: Dr. M. T. Manzuri

Winter 2004