近傍の銀河団内の棒渦巻銀河における $\mathrm{H}\alpha$ 輝線観測及び名寄望遠鏡搭載用分光撮像装置の製作

北海道大学理学部物理学科
宇宙物理学研究室 学部 4 年

中尾 光

2010 年 4 月 13 日

概要

我々の住む銀河系は渦巻銀河と考えられているが、渦巻銀河の中には銀河中心にバー構造のある、棒渦巻銀河が存在することが知られている。近年のミリ波帯での観測から棒渦巻銀河のバーにおける分子ガスの分布が 4 つに大別されることが分かってきた。その 4 つとは、(1) バー全体に分子ガスが分布しているもの、(2) バーの両端と中心部に分子ガスが集中しているもの、(3) 中心部に分子ガスが集中しているもの、(4) 分子ガスが殆ど存在しないものである。

本研究ではこの 4 つの違いが、銀河の進化段階の違いによるものなのか、それとも銀河固有の性質によるものなのかを確かめるために、異なる赤方偏移の銀河団 (A2634 と A426(ペルセウス銀河団)) における Hα 輝線の観測を行った。遠方の銀河を観測することは過去の銀河を観測することであるので、異なる赤方偏移の銀河団内の多数の棒渦巻銀河を観測することで棒渦巻銀河の年代における構造の違いが明らかにできると期待される。

この研究を発展させるために、北海道大学が名寄市に建設中の口径 1.6 m の光学赤外線望遠鏡に可視域の分光撮像装置を製作し、それを用いた分光撮像を行う予定である。この製作過程のうち、検出器の読み出し回路周辺について実施した作業及び実験についても報告する。

目次

1　はじめに

1.1　銀河の分類

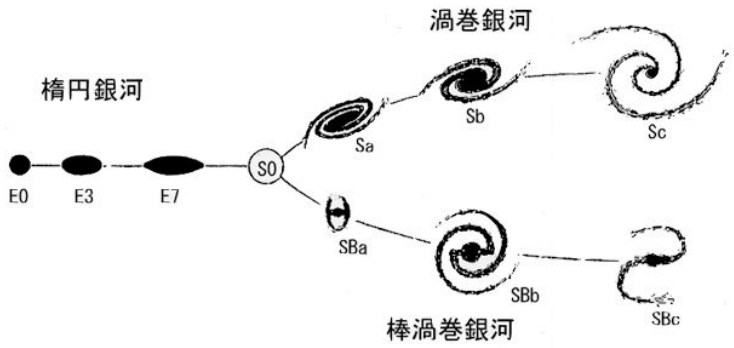


図 1.1　ハッブルの音叉図 (Hubble, E. 1936)

銀河は宇宙の基本単位であり、典型的な銀河の質量は 10^{10} M$\odot$、直径は 30 kpc 程度であるが、小さい銀河と大きい銀河は質量やサイズで 1000 倍近い差があり、その形態は多様である。この様に多様な銀河はハッブルの音叉図により大まかに分類することができ、これは銀河の分類の基礎となっている。

ハッブルの音叉図によると銀河は楕円銀河 (E) と渦巻銀河 (S) に大別され、渦巻銀河で中心に棒構造があるものは棒渦巻銀河 (SB) に分類される。また、ハッブルは楕円銀河と渦巻銀河の中間にレンズ状銀河 (S0) の存在を仮定しているが、後の観測にレンズ状銀河の存在も確認されている。この音叉図に無いものに不規則銀河 (Irr) があり、音叉図の最も右側に位置するとされている。

かつて銀河はハッブルの音叉図の左側から右側に進化すると考えられていたために楕円銀河とレンズ状銀河を早期型銀河と呼び、渦巻銀河、棒渦巻銀河、不規則銀河を晩期型銀河と呼ばれる。実際の進化は必ずしもそのように進むわけではなく、その解明のために現在も盛んに研究が行われている。

1.2　H II 領域

星間ガス内に活発に活動する大質量星の O 型星及び、B 型星がある場合、それらが放射する紫外線により星間ガスは電離される。星間ガスの成分のほとんどを水素原子が占めており電離した水素を H II と表すことから、この電離された星間ガスの領域は H II 領域と呼ばれている。また、大質量星は巨大分子雲内で形成されるため、H II 領域は巨大分子雲と隣接し星形成領域となっていることが多い。H II 領域からの放射で最も強力なものは Hα 輝線 (波長 656.28nm) であり、これは水素原子の電子軌道が量子数 n=3 から n=2 に遷移 (バルマー系列) した際の放射輝線であるが、その放射過程は電離した水素原子が高いエネルギー準位に再結合し、その後電子が下の準位 (n=3 から n=2) に遷移することで放射される。Hα 輝線の他にも [O III] 輝線 (波長 495.9 nm と 500.7 nm) や [O II] 輝線 (波長 372.7 nm と 372.9 nm) も強く放射されており、Hβ 輝線 (波長 486.1 nm)、Hγ 輝線 (波長 434.0 nm)、[N II] 輝線 (波長 654.8 nm と 658.4 nm)、[S III] 輝線 (波長 671.7 nm と 673.1 nm) の順に強く放射している。Hα 輝線は渦巻銀河の渦巻腕で特に強く観測されている。そのため H II 領域は渦巻銀河の渦巻腕に集中しており、そこでは星形成が活発に起きていると考えられている。

2　観測

表 2.1　観測時間の予定と結果

銀河団	フィルター	1 フレームの露光時間	必要な露光時間	実際の露光時間
A2634	R	300 秒	30 分	30 分
	V	300 秒	30 分	15 分
	B	300 秒	30 分	15 分
	Ha6577	300 秒	7 時間	1 時間 27 分
	Ha6737	300 秒	7 時間	1 時間 17 分
A426	R	300 秒	15 分	4 分半
	V		15 分	0 分
	B		15 分	0 分
	Ha6577	90 秒	3 時間半	30 分
	N668	90 秒	3 時間半	30 分

長野県木曽郡にある東京大学大学院理学系研究科天文学教育研究センター木曽観測所の 105 cm シュミット望遠鏡を用いて 2009 年 9 月 26 日から 10 月 3 日までの 8 夜、A426(ペルセウス銀河団) 及び A2634 の観測を行った。これらは近傍の銀河団 (赤方偏移 0.0183 及び 0.0312) であり、どちらも所属する銀河数が多い。また、赤方偏移した Hα スペクトル線の波長はそれぞれ 6683 Å と 6768 Å であり狭帯域フィルター N668 及び Ha6737 に含まれているため、Hα のオンバンドとして使用した。このフィルターで観測することで赤方偏移した Hα をとらえることができるはずである。その他に使用したフィルターは Hα のオフバンドとして Ha6577 を、また広帯域フィルターとして B、V 及び R バンドフィルターを使用した。Hα のオンバンドでの観測により、銀河団内の数 10 個の棒渦巻銀河の星形成領域の大まかな分布が明らかになり、異なる赤方偏移の銀河団を比較することでその違いの有無を確認できることが期待できる。

2.1　観測状況

この観測での必要と思われる露光時間は A2634 では 3 つの広帯域フィルターで 30 分づつ、2 つの挟帯域フィルターで 7 時間づつであり、A426 では前者が 15 分、後者が 3 時間半である。本観測は天候に恵まれず、各フィルターでの露光時間は必要な時間に届かなかった。両銀河団での各フィルターにおける露光時間は表 2.1 である。なお、露光中に曇った等で使用出来なかったフレームの露光時間は含まれていない。

2.2　解析

2.2.1　一次処理

本観測では A2634 と A426 以外に、バイアスとフラット、標準天体も観測している。これらは観測フレーム内の天体の情報以外に含まれているものを取り除くために使われる。

バイアスフレームとは、シャッターを開けず露光時間ゼロ秒の時の CCD の暗電流や読み出しノイズを反映するフレームである。

本研究でのフラットフレームはドーム内のスクリーンを一様な明るさに照らし、それを観測することで得られる CCD の感度ムラを反映するフレームである。

標準天体は絶対等級が分かっている天体であり、目的の天体と標準天体を観測することで大気による減光を補正することができるが、観測天体と同様に標準天体もいくつか観測できなかった。このためキャリブレーション

はできていない。解析ソフトは IRAF を FITS 画像の表示には ds9 を使用した。

2.2.2　一次処理の手順

表 2.1 のように露光時間がかなり不足しており、精度の高い解析はできないが、参考までに一次処理を行った。一次処理は生データからバイアスを引き、フラットで割り、各データの位置合わせをして足し合わせるという手順である。

例として、A2634 をフィルター Ha6737 で観測したデータの 1 次処理を記す。
まず、露光時間 90 秒での生データ (ファイル名は kcd151182.fits) とバイアスフレーム、フラットフレームの生データを図 2.1 に示す。フレームの左側に縦に入った線は CCD の欠損によるものである。

図 2.1　左から天体、バイアス、フラットの生データ。

バイアスフレームやフラットフレームも 1 枚ごとに誤差が生じるため、複数毎の平均を取って使用する。A2634 を観測した日は 20 枚のバイアスを撮っており、その平均を取ったバイアスフレームは図 2.2 である。1 枚のフレームよりもムラが無くなっていることが分かる。

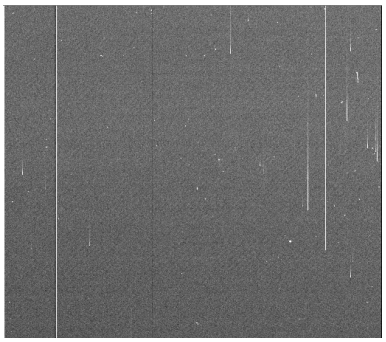


図 2.2　平均したバイアスフレーム

次にフラットフレームの平均を取るが、フラットフレームの生データにはバイアスフレームの効果が付加されているので、フラットフレームの生データからバイアスフレームを引き、その後で平均を取った。

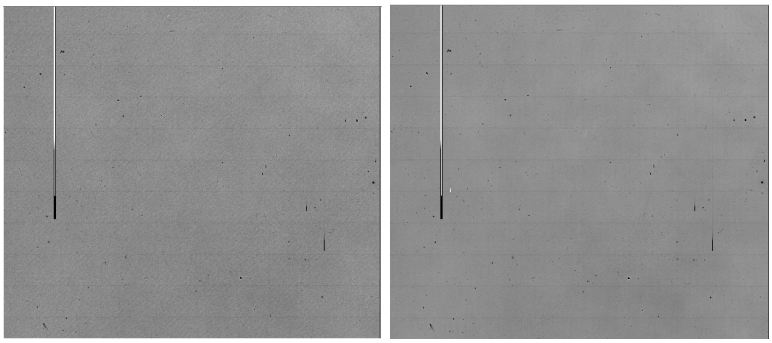


図 2.3　(左) フラットからバイアスを引いたもの。(右) それらを平均したフラットフレーム

A2634 の生データフレームについては、まずバイアスフレームを引き、その後でフラットフレームで割る。生データとバイアスフレームを引いた後、フラットフレームで割った後の fits 画像を図 2.4 に示す。10 月 3 日に取

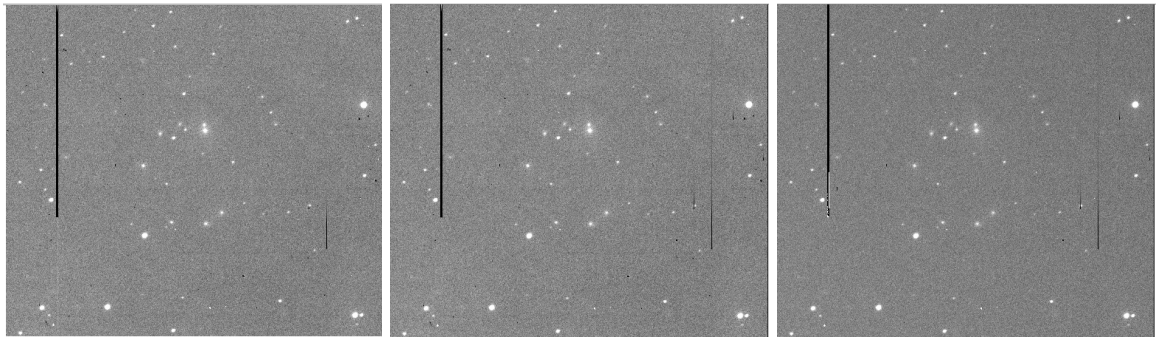


図 2.4　左から生データ、バイアスを引いたもの、フラットで割ったもの。

得したフィルター Ha6737 での A2634 のデータフレーム数は 19 枚 (露光時間 32 分) であり、バイアスとフラット処理を施したそれら全てのフレームの天体の位置を合わせて足し合わせたものが図 2.5 である。CCD の欠損部が広がっているのは位置の異なったデータフレームを重ね合わせたためであり、本来はこのような明らかな欠損部を取り除いてから足し合わせるが、今回は行っていない。フィルター Ha6737 を用いて観測した A2634 の全てのデータフレームを足し合わせた最終結果は図 2.7 である。

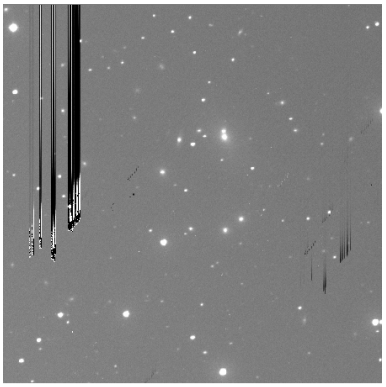


図 2.5　10 月 3 日の Ha6737 での A2634 のデータフレームを全て足し合わせたもの

2.3　結果

上記の方法での 1 次処理を 8 夜の観測で取得した全てのデータに行い、その結果 fits 画像図 2.6 と図 2.7 を得た。 参考までに B、V、R の 3 つのフィルターでの観測結果を用いて、3 色合成した図 2.8 載せる。

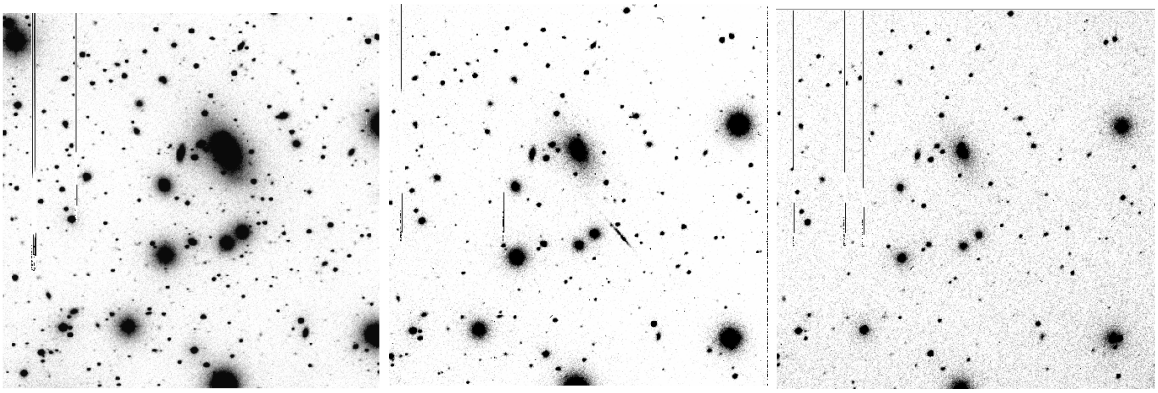


図 2.6　左から A2634 の B、V、R バンド画像。

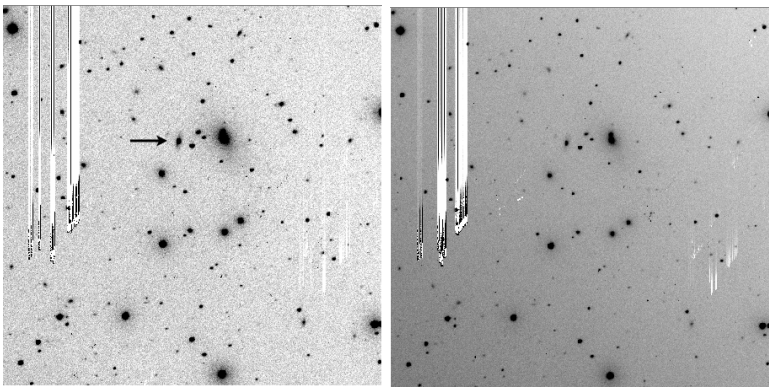


図 2.7　A2634 の (左)Ha6737、(右)Ha6577 バンド画像。矢印で示した銀河の拡大図は図 2.10 参照。

図 2.8　B、V、R バンドの 3 色合成画像

A426 においても同様の一次処理をして、図 2.9 を得た。

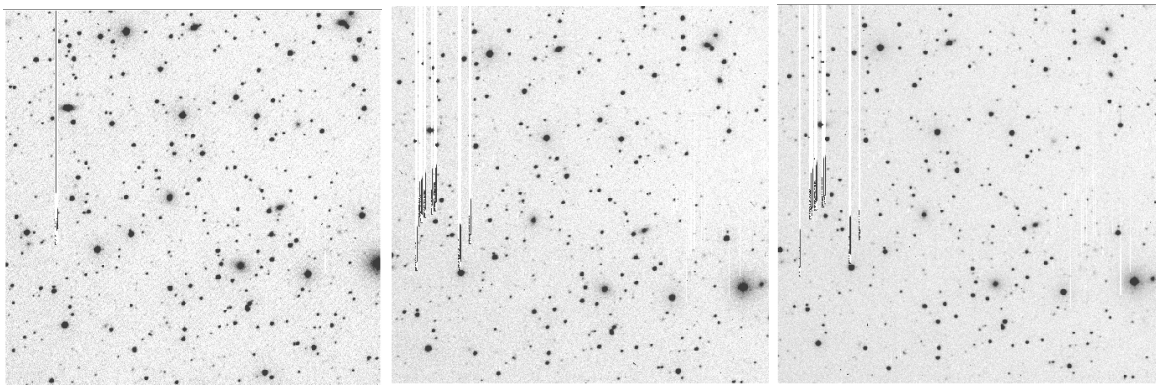


図 2.9　左から A426 の R、n688、Ha6577 フィルター画像。

2.4　考察

Hα 輝線のオンバンドで観測した図 2.7 及び図 2.9 に渦巻腕構造がはっきりと見える銀河は見られなかった。しかし、A2634 を Ha6737 フィルターで観測した結果と Ha6577 フィルターで観測した結果を比べると、Ha6737 フィルター観測したものでは、図 2.7 に矢印で示した銀河 (拡大したものは図 2.10 の右図) が上下に広がって見える。Ha6737 フィルターでは Hα 輝線を観測しているので、この銀河は渦巻銀河である可能性があるが、Dressler et al (1980) はこの銀河をレンズ状銀河 (S0) に分類している。この食違いについても、露光時間を増やすことでより詳細な議論ができると思われる。

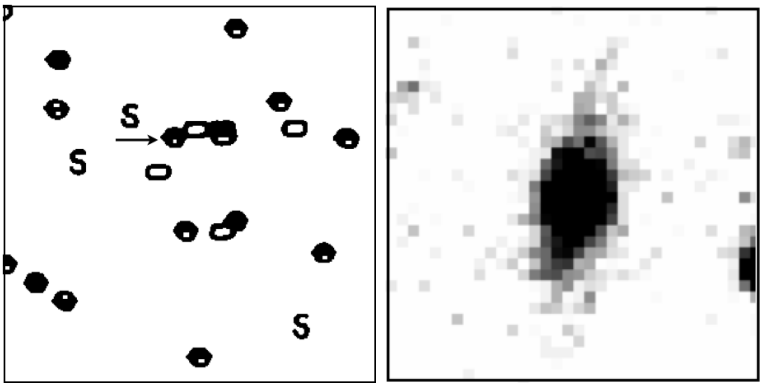


図 2.10　(左)Dressler et al (1980)A2634 の銀河の分類。矢印で示した黒丸は渦巻銀河 (S0)、白丸は楕円銀河、S は渦巻銀河である。(右) 図 2.7 の矢印で示された銀河の拡大図。

3　分光撮像装置の製作

3.1　概要

現在、北海道大学では宇宙理学専攻惑星宇宙グループが中心となり北海道名寄市の北海道立サンピラパーク内の星見の丘に 2010 年 4 月開館予定の天文台 (きたすばる) に口径 1.6m の光学赤外線望遠鏡を製作している。望遠鏡は 2011 年 3 月完成予定であり、おおよそ東経 142 度 28 分 58 秒、北緯 44 度 22 分 27 秒に位置している。天文台には、1.6m 光学赤外線望遠鏡の他、口径 50、40、25cm の望遠鏡やプラネタリウムが設置されている。

自大学の望遠鏡は、他の光学赤外線望遠鏡と比べると長い露光時間が必要な観測を行うことができる。1.6m 光学赤外線望遠鏡ではそれを利用して近傍の銀河の星形成史を探りたい。しかし、この望遠鏡には惑星観測用の観測装置は搭載される予定だが、銀河系内の恒星や系外銀河を研究できる装置がなかったので製作する。製作は東京大学及び神戸大学のグループと共同で行い、我々の研究室は分光撮像装置のエレクトロニクスを製作する。分光器製作には時間が掛かるため、まずは撮像部分を先行して製作しその後で分光器を追加する予定である。製作に伴い、2009 年 11 月と 2010 年 1 月にそれぞれ 1 ヶ月間の合計 2 ヶ月間を東京大学大学院理学系研究科天文学教育研究センターに滞在し読み出し回路の製作を行い、分光撮像装置製作に必要な技術を教えていただいた。

4　CCD と読み出し回路

4.1　CCD の原理

現在の天文学における光検出器はほとんどが CCD(charge coupled device、電荷結合素子) であり、その量子効率は波長にもよるが 90 % 以上を達成しており、それ以前の写真乾板の量子効率 2 % と比較するとかなり感度が良く、観測結果を容易にデジタル化できる等の利点がある。今回製作する分光撮像装置の検出器にも CCD を使用する。

CCD は半導体の素子が複数集まってできている。半導体に光が入射すると光電効果によって、半導体内に伝導電子とホールのペアができて、CCD では半導体につけた電極に +10V 程度の電圧をかけることで生じた電子を保持することができる。CCD の素子は図 4.1 のように 1 ピクセルに 3 つの電極がついており (これを 3 層式 CCD という) ピクセル内での光電効果で発生した伝導電子は高電圧の掛かった電極に集められて、その隣の電極付近の半導体はホールが無くなるため絶縁体として機能する。CCD に溜まった電荷の転送には図 4.3 のように電極の電圧を変更させることによって、電極に溜まった電荷は隣の電極に順番に転送されていく。縦の列 (図 4.2 の (b)) のピクセルは、絶縁層で区切られていてピクセル間での電荷の移動は起きない。図 4.1 で右方向に転送された電荷はその後下方向に転送されて、それが終わると次の列の転送が始まる。

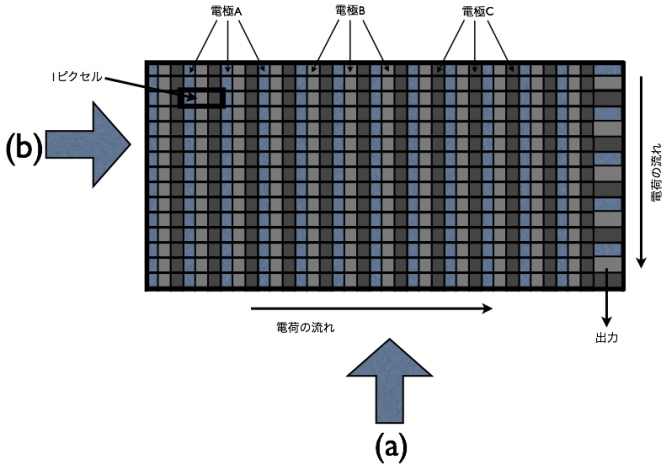


図 4.1　CCD のピクセル図

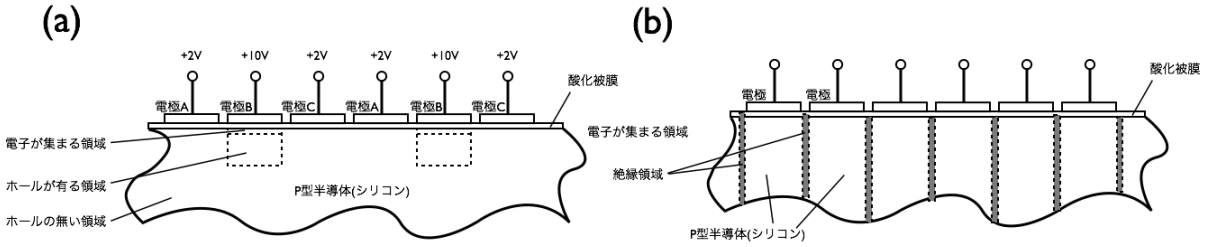


図 4.2　図 4.1 を (a) 方向から見た電極 (左)、(b) 方向から見た電極 (右)

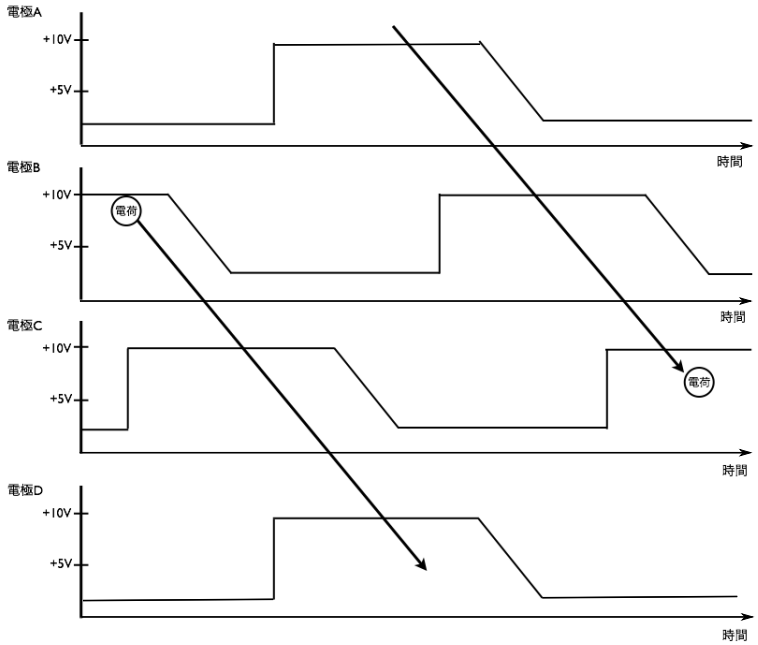


図 4.3　電荷を転送する時の各電極での電圧の変化

4.2　読み出し回路

CCD はそれ自体では動作しないため CCD を観測に用いるためには、CCD を制御する読み出し回路の製作が必要である。今回製作する読み出し回路では、回路を 4 つのボードに分割している。その 4 つのボードはマザーボード、IF(インターフェイス) ボード、DRV(ドライバー) ボード、ADC(AD コンバータ) ボードの 4 つである。マザーボードは読み出し回路の母体となるボードであり、各ボードをつなぎ、電源を供給するボードである。また、IF ボードは計算機と DRV ボード、ADC ボードと計算機の通信を行うボード、DRV ボードは CCD を制御するボード、ADC ボードは CCD からの信号を処理するボードである。現在製作している、IF ボードと DRV ボードに関する詳しい説明は後述してあるが、回路全体の配線は図 4.4 のようになっている。

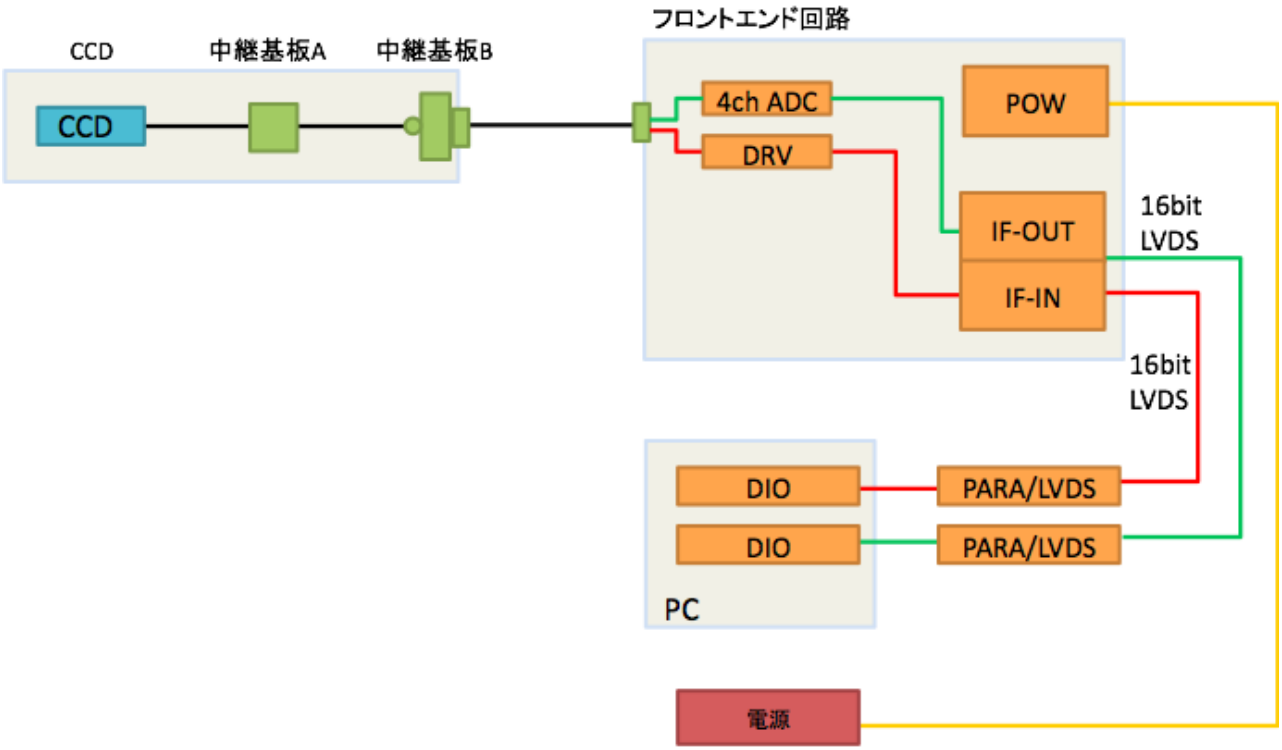


図 4.4　読み出し回路の配線図

4.2.1 データ通信

デジタル信号の通信にはパラレル通信とシリアル通信がある。この２つの通信の違いは、例えば 16 ビットのデータを転送する場合に 16 本の信号線で 16 ビットを同時に送信する通信がパラレル通信、その時の信号がパラレル信号であり、16 ビットの信号を 1 本の信号線で送信する通信がシリアル通信、その時の信号がシリアル信号である。ケーブルでの通信をパラレル通信でおこなうと、ケーブル間の浮遊容量によるノイズや、高周波の通信や長いケーブルでの通信では各信号の同期を取るのが難しくなるが、シリアル通信ではこの問題は解決され、またケーブルの幅やコネクタのサイズやピン数を減らすことができるため、コストを抑えることもできる。今回のボードでは制御室の計算機から望遠鏡までの通信をシリアル通信で行うために、PC 内でのパラレル信号を DS90UR241 という IC を使いシリアル信号に変換しそれを CCD の読み出し回路まで転送する。また、読み出し回路内ではパラレル信号を使用するので、シリアル信号を DS90UR124 という IC でパラレル信号に変換している。

4.2.1.1 シリアライザとデシリアライザ

DS90UR241 のシリアライザと DS90UR124 のデシリアライザのチップセットは 24 ビットのパラレルデータをスループット (単位時間あたりの処理能力)120Mbps 〜 1.03Gbps のクロック情報を埋め込んだ単一シリアル LVDS データで転送するチップセットである。

DS90UR241/124 の I/O(入力/出力) には LVDS(低電圧差動信号) を採用しており、低消費電力で低ノイズな転送方式により、最長 10m の STP(シールドツイストペア) ケーブル上で 5MHz 〜 43MHz のクロック速度でデータを転送できるように設計されている。

デシリアライザは外付けのリファレンスクロックを使用せずにデータパターンに関係なくシリアライザと同期するため、特別なパターンや同期コードが必要ない。デシリアライザは埋め込まれたクロック信号を抽出し、入力データ列からデータ品質を確認することによってクロックおよびデータを回復した後、データをデシリアライズし、受信クロック情報から PLL(入力信号や基準周波数と出力信号の周波数を一致させる電子回路) のロック状態を決定してロックが発生したときに LOCK 出力を HIGH にアサート (出力を有効に) する。

4.2.1.2 初期化とロックメカニズム

DS90UR241/124 でデータの送受信を行う前に、これらのデバイスを初期化しなくてはならない。初期化とはシリアライザとデシリアライザの PLL 同士を同期させることで、まずシリアライザを入力クロックソースにロックし (段階１)、次にデシリアライザをシリアライザに同期させる (段階２)。

段階 1:シリアライザとデシリアライザの各チップに電源電圧 V_{DD} が供給されると、チップに内蔵された電源オン制御回路によって各出力は TRI-STATE(出力が HIGH、LOW、ハイインピーダンスの 3 つの状態) となり、内部回路はディスエーブル (無効に) される。V_{DD} の電圧値が V_{DD}OK(〜 2.2V) に達すると、シリアライザの PLL はクロック入力に対してロックを開始する。シリアライザ側は PLL が送信クロック TCLK にロックするまでの間、出力は TRI-STATE のままで、TCLK にロックするとデータパターンの出力準備が完了する。デシリアライザは PLL がシリアルデータ列に埋め込まれたクロック情報にロックしている間は、出力を TRI-STATE 状態に保ち、またデシリアライザの LOCK ピンは、$R_{IN}\pm$(レシーバ LVDS 非反転/反転入力) ピンで受信したランダムなデータまたは、SYNC(同期) パターンに PLL がロックするまでの間は LOW を出力する。

段階 2:デシリアライザの PLL は、シリアライザからの特別なパターンを使用せずにデータ列に対するロックができるため、初期化の段階ではシリアライザからデシリアライザに対してランダムデータパターン (非繰り返しパターン) を自動的に出力する。埋め込みクロックと CDR 回路 (クロックデータリカバリ回路) は入力ビット列にロックして、クロックビットとデータビットを分離し、分離されたクロックの立ち上がりのエッジを検出することで、デシリアライザはシリアライザからのランダムデータパターンへのロックを完了する。埋め込みクロックに

対してデシリアライザの CDR がロックに成功した時点で、デシリアライザの LOCK ピンは HIGH になり、出力ピンに出力される RCLK データ (パラレルインターフェイスのクロックデータ) が有効になり、LOCK 信号は出力ピンに有効データが現れるタイミングに同期する。以上が初期化の過程である。

4.2.1.3　データ転送

チップの初期化後はデータの送受信が可能になる。送信される 24 ビットのデータは 24 本の入力ピンを使用してシリアライザに入力され、そのデータはシリアライザの TCLK により取り込まれる。この時、1 つの LVDS シリアルデータ列には CLK0、CLK1、DCA、DCB の 4 つのオーバーヘッドビットが付加されており、CLK0 は常に LOW に、CLK1 は常に HIGH にすることで、CLK0 ビットと CLK1 ビットはシリアルデータ列の両端に付加されてシリアルデータ列の埋め込みクロック情報となる。また、DCA ビットは埋め込みデータ列のデータ品質を確認するために使用され、DCB ビットは信号ラインの DC バイアスを最小にするための、DC バランス制御ビットとして使用される。シリアライズされたデータと埋め込みクロックおよび制御ビット (24+4 ビット) は TCLK 周波数の 28 倍でシリアライザから出力され、デシリアライザはシリアライザからの入力をロックすると LOCK ピンが HIGH になり、有効なデータと復元したクロックを同時に出力することでデータの送信が完了する。今回製作する読み出し回路では 16 ビットのデータ通信を行っている。

4.2.1.4　LVDS ボード

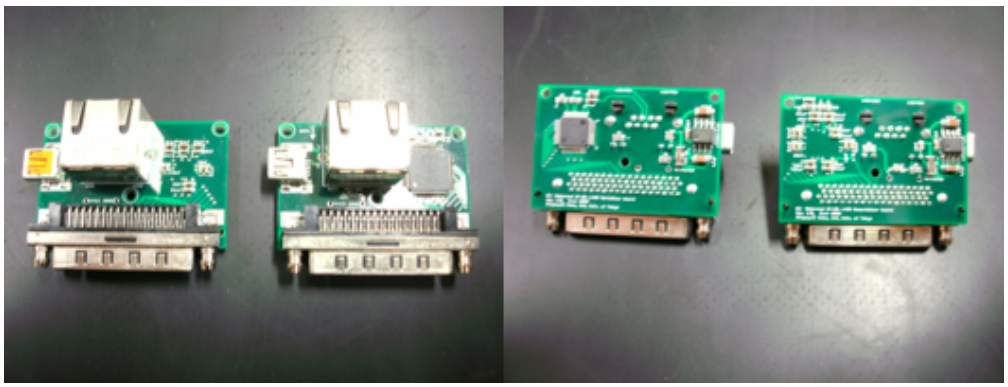


図 4.5　LVDS ボードの 1 層目 (左) 及び 4 層目 (右)。また両面において左側がシリアライザ、右側がデシリアライザである。

計算機と読み出し回路とのシリアル通信を行うために、計算機側には LDVS ボードがつけられている。このボードには上記の IC チップ DS90UR241 と DS90UR124 がついており、IF ボードと 16 ビットシリアル LVDS 通信を行っている。また、このボードは 1 層目及び 4 層目に IC 等のパーツが取り付けられており、2 層目はグラウンド層、3 層目は電源層となっている。

4.2.2 IF ボード

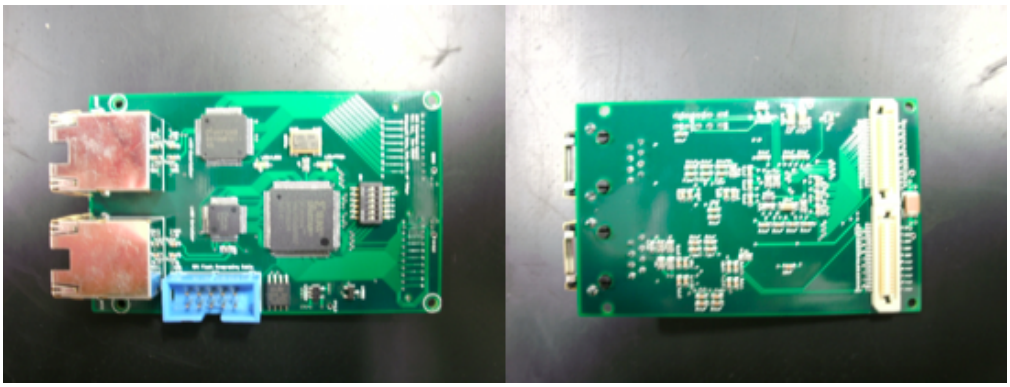


図 4.6 IF ボード 1 層目 (左) 及び 4 層目 (右)。

IF ボードは LVDS ボードと同様にパラレルシリアル変換ボードであるが、DRV ボードへパラレルデータ信号を転送し、ADC ボードからの 16 ビットのシリアル信号 (50MHz 同期) をパラレル信号 (10MHz 同期) に変換して IF ボード内のシリアライザに転送する役割も持っている。ADC ボードからの信号を処理しているのは FPGA(Field Programmable Gate Array) という IC である。

4.2.2.1 IF ボード内の FPGA

FPGA は利用者が目的に応じてプログラムできる IC である。製造段階で用途が決められている一般的な IC(ASIC) は完成後に用途を変更することができない。そのため動作は非常に高速であるが、目的の動作をする IC が必要な場合は特注で作る必要があり、それはとても高価である。それに対して FPGA は完成後に利用者がプログラムすることができるため、IC を特注する必要がなく比較的安価である。

4.2.3 DRV ボード

DRV ボードは CCD を制御するために必要な電圧を供給するボードである。CCD は製品により差はあるが、制御するために 20 V 程度と-15 V 程度の高電圧を必要とする。しかし、読み出し回路内のほとんどの IC 類は 3 V 程度で動作する仕様なので、CCD 専用に電源を供給する必要がある。PC からの CCD を制御するコマンドは 2 進数で 1(HIGH) と 0(LOW) の決まった電圧しか出力できない。そのため CCD を制御する電圧に変換しなくてはいけない。DRV ボードはそれを行っている。

4.2.3.1 オペアンプ

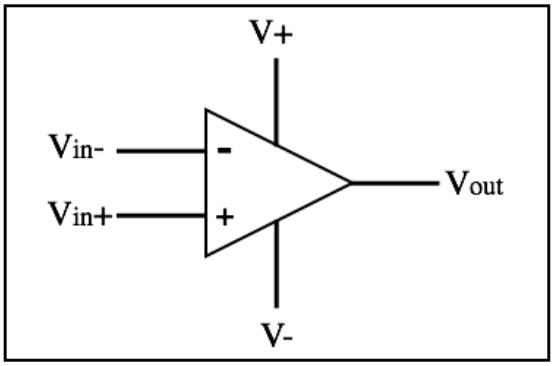


図 4.7 オペアンプの図記号

DRV ボード内での電圧の変更にはオペアンプを用いている。オペアンプは Operational Amplifier(演算増幅器) のことで、アナログ回路での増幅器である。CCD からの信号は CCD に入射した光量に比例する電圧を示すが、各ピクセルでの電圧の差は微弱であるため、正確に扱うことが困難である。この微弱な信号はオペアンプにより増幅することで正確に扱えるようになる。

一般的なオペアンプの図記号は図 4.7 である。オペアンプに電源を供給する端子は V+ と V-のように 2 つあり、用いる記号や使用する電圧は様々であるが、図記号では必要なとき以外は省略される場合が多い。オペアンプの出力端子は 1 つであるが、入力端子は + と-の 2 つがあり、+ 端子からの入力の場合は出力端子から + の出力が得られ、-端子からの入力の場合は-の出力が得られるので、+端子からの入力を非反転入力、-端子からの入力を反転入力という。オペアンプは多数のトランジスタや抵抗、コンデンサーの集まりであるが、小型で低価格化されているため抵抗のように 1 つの電子部品として扱うことができる。

4.2.3.2 オペアンプの特性

理想的なオペアンプの特性は以下の 3 つである。

- オープンループゲイン A が無限大
- 入力インピーダンスが無限大
- 出力インピーダンスがゼロ
- 出力は入力電圧の差のみに比例する。つまり、$V_{out} = A(V_{in+} - V_{in-})$。

オープンループゲインとは図 4.8 の様に入力端子と出力端子を抵抗等を挿んで繋ぐ”ネガティブフィードバック”をかけない時のオペアンプのゲインである。つまり、ネガティブフィードバックをかけない場合はどんなに小さな電圧でも、電源電圧まで増幅される。しかし、実際のオペアンプのオープンループゲインは有限で定数ではないために出力電圧が揺らいでしまう。そのため普通はネガティブフィードバックをかけて、出力電圧の一部を入力端子に戻しゲインを安定させる方法がとられる。

次に入力インピーダンスが無限大というのは、オペアンプの入力端子に電流が流れないことを表しており、出力インピーダンスがゼロというのは、出力端子の抵抗値が 0 Ω であることに相当する。

4.2.3.3　イマジナリーショート

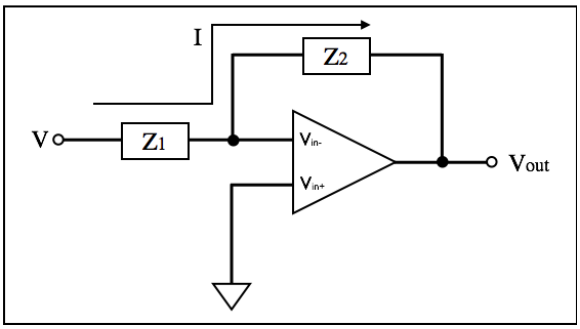


図 4.8　非反転入力が接地されている反転増幅回路

上記のオペアンプの特性からイマジナリーショートという、オペアンプの重要な関係が得られる。それは図 4.8 のような非反転入力が接地されている反転増幅回路を考えることで示される。この回路で各インピーダンスに流れる電流 I は

$$I = \frac{V - (V_{in-} - V_{in+})}{Z_1} = \frac{(V_{in-} - V_{in+}) - V_{out}}{Z_2} \tag{4.2.1}$$

非反転入力が接地されているので $V_{in+} = 0$ であるから

$$I = \frac{V - V_{in-}}{Z_1} = \frac{V_{in-} - V_{out}}{Z_2} \tag{4.2.2}$$

オペアンプの特性より $V_{out} = A(V_{in+} - V_{in-}) = -AV_{in-}$ なので代入すると

$$\frac{V - V_{in-}}{Z_1} = \frac{V_{in-} + AV_{in-}}{Z_2} \tag{4.2.3}$$

$$\tag{4.2.4}$$

となり、これより

$$V_{in-} = \frac{VZ_2}{(1+A)Z_1 + Z_2} \tag{4.2.5}$$

$$\tag{4.2.6}$$

となる。A の値は通常 10^4 以上であり、他のインピーダンスに対してかなり大きいので $(A \gg Z_1, A \gg Z_2)$

$$V_{in-} = \frac{VZ_2}{(1+A)Z_1 + Z_2} \approx 0 \tag{4.2.7}$$

が成り立つ。この非反転入力端子と反転端子の電位差が理想的には等電位になることがイマジナリーショートである。

4.2.4　IF ボード試験

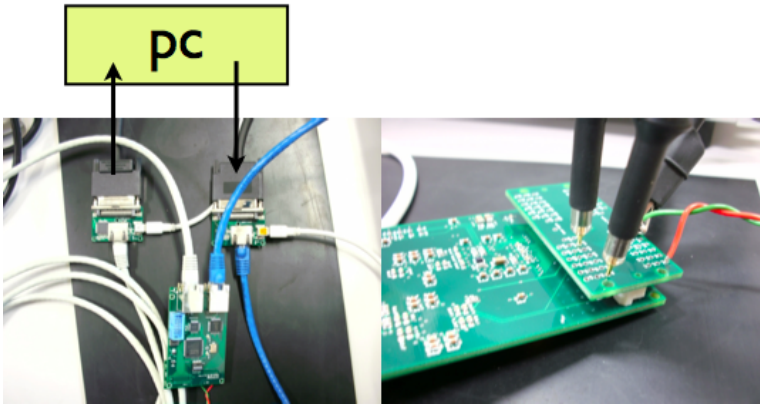


図 4.9　(左) ボードの配線、(右) 端子にオシロスコープのプローブをあてている。

FPGA を使用するためにはプログラムを書き込む必要があるが、書き込みの時点でプログラムが正常に書き込まれたかを判断することはできない。そのため IF ボードには、FPGA のプログラムがロードされている時に HIGH になる DONE ピンに LED を取り付けてプログラムが正常に書き込まれているかが分かるようしている。また、LVDS ボードからの信号をデシリアライザが受け取れているかを確認するために LOCK ピンにも LED を取り付けてある。今回行った実験の目的は IF ボードの FPGA とデシリアライザが正常に動作するかをチェックすることである。

試験は図 4.9 のように PC、LVDS ボード、IF ボードを接続して、PC からサンプルソースファイルを出力した時のデシリアライザ側の端子での波形の測定と、FPGA とデシリアライザに接続された LED 周りの電圧の測定を行った。設計ではデシリアライザ側のコネクタ端子 a1 から a20 では a1 に 10MHz の同期クロック (RCLK)、a18 に GATE 信号 (GATE)、a19 に LOCK 信号 (LOCK) が検出されて、a2 から a17 までに 16 ビットのデータ信号が出力されるはずである。また、a20 は使用していない。GATE 信号とは、PCI-EX ボードがデータを出力されている間は電圧が LOW になるように設定されている信号である。サンプルソースファイルは 1010101010101010 と 0101010101010101 の 16 ビットパラレル信号を交互に出力して試験を行った。

4.2.4.1 IF ボード試験結果①

デシリアライザ側コネクタの各端子に設計通りの信号が出力されているかを試験し、表 4.1 に示す結果を得た。なお、この時の電圧の測定はオシロスコープの波形を目分量で測っている。

本試験で確認された設計と異なる動作は、a17 端子と GATE 信号でデータ送信直後に約 680 msec の間電圧が HIGH になったことであるが、これは PCI-EX ボードの仕様で前回の通信での最後のビットの情報を、次の通信時に先頭のデータが来るまで出力し続けることが原因と分かった。つまりデータ出力端子 a3, a5, a7, a9, a11, a13, a15, a17 と GATE 信号は通信の最後のビットが 1(HIGH) であるので、次の通信の際に LOCK 信号が HIGH になってからデータが出力されるまでの約 680 msec 間電圧が HIGH になることで説明がつく。

その他のどの端子でもおおむね設計通りの波形が得られ、同期クロックと各波形とで通信に影響する様な時間差は確認されなかった。a20 端子で検出された 70 mV、5 MHz の矩形波は他の端子の波形に誘導されて検出されたものであるが、電圧が低いので IF ボードに悪影響を与えることは無いと考えられる。試験中に測定した波形の一部を図 4.10 と図 4.11 に載せる。

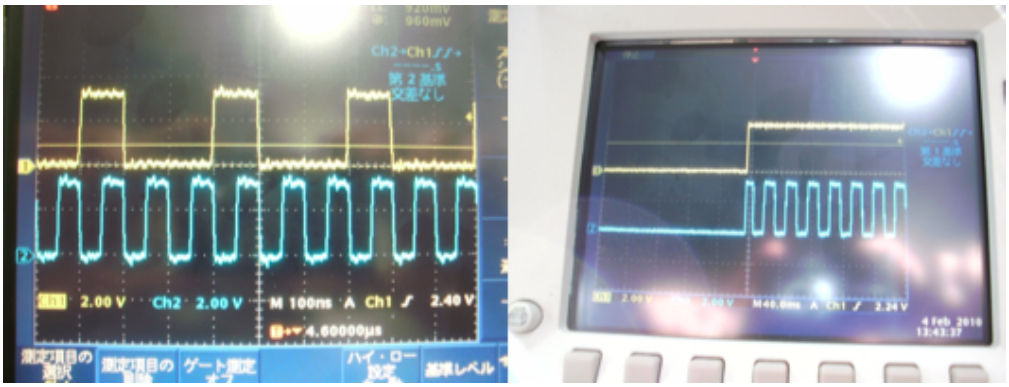


図 4.10　(左) 同期クロック (青) と a2 端子での 100100... の波形 (黄) の比較、(右) 同期クロック (青) と LOCK 信号 (黄) の立ち上がり。

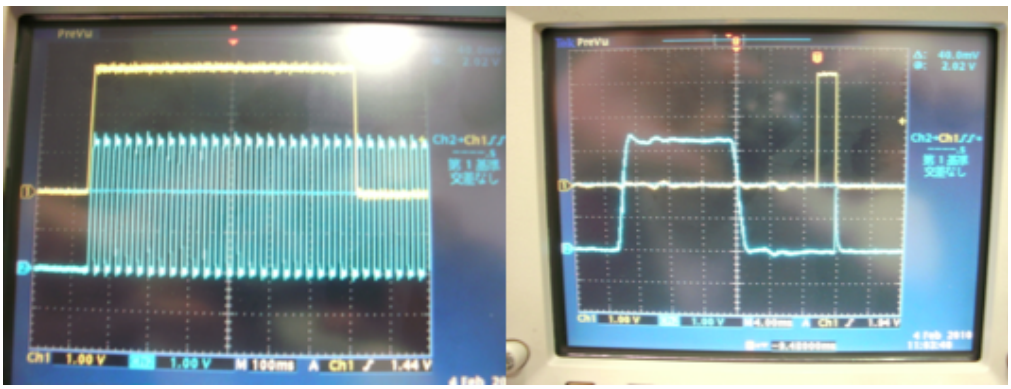


図 4.11　(左) 同期クロック (青) と GATE 信号 (黄) の立ち上がり。(右) 通信の終わり。

表 4.1　IF ボード試験結果①

端子	測定電圧	測定周波数	1 ビットの秒数	予想される波形
a1(RCLK)	3.3 V	10 MHz		10 MHz の矩形波
a2	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a3	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a4	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a5	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a6	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a7	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a8	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a9	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a10	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a11	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a12	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a13	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a14	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a15	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a16	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a17	3.3 V	5 MHz	100 nsec	5 MHz の矩形波
a18(GATE)	3.3 V	通信の始めに 680 msec 終わりに 2 msec、HIGH になった		データ転送の最後に 一瞬だけ HIGH になる。
a19(LOCK)	3.3 V	データ転送中 HIGH		データ転送中 HIGH
a20	70 mV	5 MHz	100 nsec	何も検出されない

4.2.4.2　IF ボード試験結果②

表 4.2　IF ボード試験結果②

		電圧 (V)	電流 (mA)
FPGA 側			
①	DONE 信号	2.698	
②	1 kΩ 抵抗	0.16	0.16
③	LED	1.78	0.16
LVDS 側			
④	LOCK 信号	3.19	
⑤	1 kΩ 抵抗	1.32	1.32
⑥	LED	1.87	1.32
IF ボード全体			
	通信前	3.3	98
	通信後	3.3	113

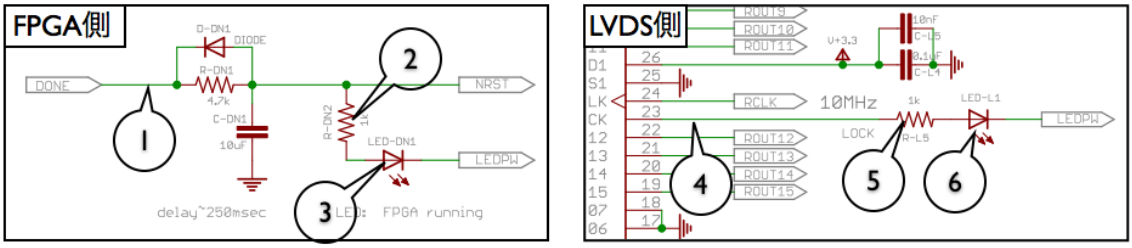


図 4.12　測定箇所の回路図

次に LVDS とデシリアライザに接続された LED 周辺の電圧、電流値の測定と IF ボード全体の電圧、電流値の測定を行った。測定した箇所は図 4.12 及び表 4.2 を参照されたい。

FPGA の DONE 信号と LVDS の LOCK 信号は共に 3.3 V を出力するはずであるが、DONE 信号の電圧は約 0.6 V も小さかった。この電圧降下の原因は FPGA の出力機構にあった。FPGA 内の出力では図 4.13 のように、リファレンス電圧にリファレンス抵抗が接続されている。これにより、LED に電流が流れるとリファレンス抵抗により電圧降下が起きることが分かった。

DONE 信号は 3.3 V のまま FPGA 内の NRST 端子に接続しなくてはいけないので、この電圧降下を防がなくてはならないが、そのためには LED の前にオペアンプを設置して NRST 端子側の回路に電流が流れないようにするか、FPGA のプログラムを変更してリファレンス抵抗を外し、LED と 1 kΩ の抵抗を 4.7 kΩ の抵抗よりも FPGA 側に接続しなくてはならない。どちらの場合も IF ボードを作り直す必要があるが、LED はプログラムのロードを確認するためで、IF ボードの動作には関係しないので LED と 1 kΩ の抵抗は回路から外すことにした。また、LVDS 側の LED に電流が流れすぎて眩しかったので 1 kΩ の抵抗を 4.7 kΩ に変更して、再度試験を行った。

結果は表 4.3 のようになった。FPGA 側の 1 kΩ の抵抗と LED を外すことで、DONE 信号の電圧は約 3.3 V に戻り、LVDS 側の LED の電流値も下がり明るさも改善された。

表 4.3　変更後の結果

		電圧 (V)	電流 (mA)
FPGA 側			
①	DONE 信号	3.28	
②	なし		
③	なし		
LVDS 側			
④	LOCK 信号	3.26	
⑤	4.7 kΩ 抵抗	1.462	0.31
⑥	LED	1.798	0.31
IF ボード全体			
	通信前	3.3	96
	通信後	3.3	108

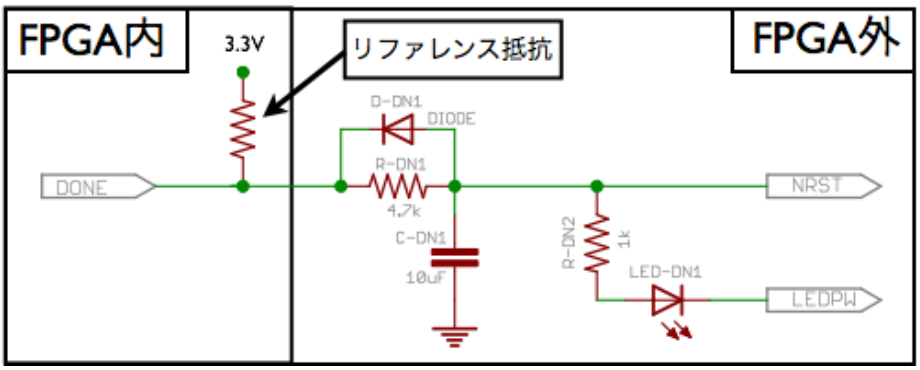


図 4.13　FPGA 内のリファレンス抵抗

4.2.5　DRV ボード抵抗値設計

4.2.5.1　オペアンプのゲイン

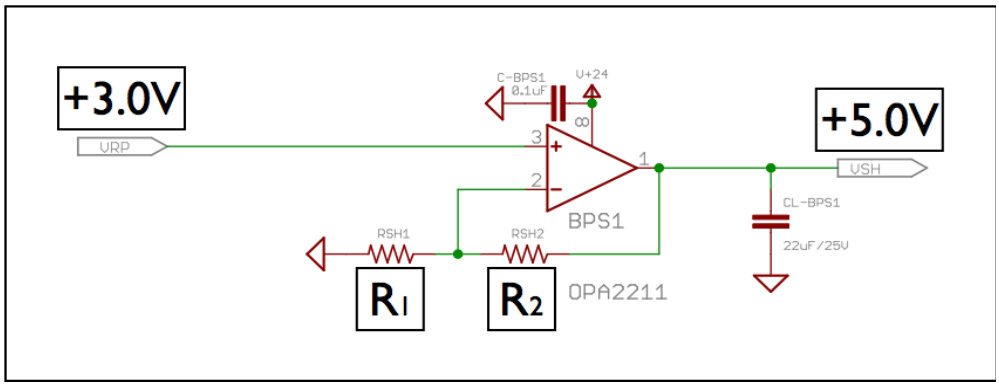


図 4.14　DRV ボード内のオペアンプの回路図。

本研究では DRV ボード内のオペアンプのゲインを決める抵抗値を求めた。計算方法は図 4.14 のオペアンプで 3 V を 5 V に増幅させる場合で考えた。オペアンプのイマジナリーショートより非反転入力端子と反転入力端子の電位差が無いと考えると、抵抗 R_1 には電圧 V_{in} が掛かる。抵抗 R_1 及び抵抗 R_2 に流れる電流が I とした時に各抵抗における電圧はオームの法則より

$$IR_1 = V_{in} \tag{4.2.8}$$

$$I(R_1 + R_2) = V_{out} \tag{4.2.9}$$

上の式 (4.2.9) を R_1 で割ると

$$I\left(1 + \frac{R_2}{R_1}\right) = \frac{V_{out}}{R_1} \tag{4.2.10}$$

$$1 + \frac{R_2}{R_1} = \frac{V_{out}}{V_{in}} \tag{4.2.11}$$

オペアンプのゲイン G は $G = \frac{V_{out}}{V_{in}}$ なので

$$G = 1 + \frac{R_2}{R_1} \tag{4.2.12}$$

となる。これより、例えば入力電圧 3 V を出力電圧 5 V で出力する場合 (ゲインが $\frac{5}{3}$ の場合) の抵抗値は式 (4.2.12) より

$$\frac{R_2}{R_1} = \frac{2}{3} \tag{4.2.13}$$

となり

$$R_2 : R_1 = 2 : 3 \tag{4.2.14}$$

となれば良い。これを満たす抵抗値は無限にあるが、実際は存在する抵抗値の抵抗を使う必要があり、この回路には $\frac{V_{out}}{R_1 + R_2}$ の電流が流れるため、ふさわしい抵抗値は限られる。DRV ボードにはこのオペアンプが 30 個 (IC の OPA2211 内には 2 つのオペアンプが入っているため、OPA2211 自体は 15 個) 使用されている。1 つのオペアンプに流れる電流はなるべく小さくするのが望ましいが、電流を小さくするために抵抗値が大きくなるとその分熱雑音が大きくなるので回路全体のバランスを取らなくてはいけない。今回は 1 つのオペアンプに平均 1mA の電流

表 4.4　オペアンプのゲイン及び流れる電流値

入力電圧 (V)	目標出力電圧 (V)	予定出力電圧 (V)	電流 (mA)	R_1 (kΩ)	R_2 (kΩ)
+3	+5	+5	1	3	2
+3	+12	+12	0.3	10	30
+3	+14	+14	1	3	11
+3	+22	約 +21.871	約 0.487	6.2	39
-3	-5	-5	1	3	5
-3	-6	-6	1	3	3
-3	-7	-7	約 1.1	2.7	3.6
-3	-8	-8	1	1.8	3
-3	-10	-10	1.25	2.4	5.6

が流れるように抵抗値を決定した。

V_{in}=3 (V)、V_{out}=5 (V) の場合は式 (4.2.8)、式 (4.2.9)、式 (4.2.14) より

$$R_1 = 3 \text{ (k}\Omega\text{)} \tag{4.2.15}$$

$$R_2 = 2 \text{ (k}\Omega\text{)} \tag{4.2.16}$$

となる。

今回使用する抵抗は進工業社製の、超高精度・高信頼金属皮膜チップ抵抗器 (RG2012 シリーズ、抵抗値許容差 0.05%) の抵抗値 2 kΩ と 3 kΩ とした。この計算で求まる抵抗値の抵抗が無い場合は存在する抵抗値で流れる電流値が 1mA に近くなる抵抗値に決定している。各オペアンプに対する抵抗値と電流値は表 4.4 に示す。

4.2.5.2　電圧モニター

DRV ボードのもう 1 つの役割は、CCD にかかる電圧を監視することである。何らかの原因により CCD に許容範囲以上の電圧が掛かってしまうのを防ぐために、DRV ボードには MAX6458 という電圧を監視する IC を載せてある。

MAX6458 はウィンドウ検出用の 2 つのコンパレータ (過電圧用と低電圧用) と、監視される入力電圧が可変電圧ウィンドウ内にあるかを示す 1 つの出力を備えた、電圧モニターである。

MAX6458 の検出ウィンドウ幅は下限値が 1.167 V、上限値が 1.228 V に設定されており、監視される入力電圧がこのウィンドウ内に収まっている時は、MAX6458 の出力端子のアサートが HIGH になり電流が流れるが、上限値を上回るか下限値を下回ると、出力端子のアサートは LOW になり電流は流れなくなる。MAX6458 は 3 つの抵抗を図 4.15 のように接続し、電圧降下させることで任意の電圧を監視することができる。

今回使用する DRV ボードでは、CCD を制御する電圧 +24 V と-15 V から ±1 V 前後を超える電圧では電流が流れないように電圧を監視する。そのために電圧が MAX6458 のウィンドウに収まるように 3 つの抵抗値を決定した。

4.2.5.3　+24 V Voltage Monitor の場合

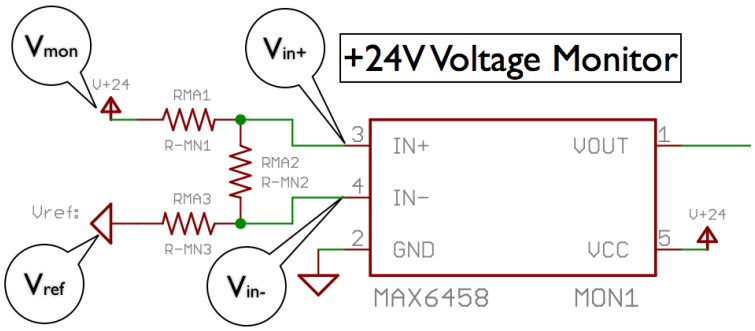


図 4.15　MAX6458 の回路図

各抵抗値と電圧の関係は以下の式で表される。

$$\begin{cases} V_{in+} = (V_{mon} - V_{ref}) \left(\dfrac{R_2 + R_3}{R_1 + R_2 + R_3} \right) + V_{ref} \\ V_{in-} = (V_{mon} - V_{ref}) \left(\dfrac{R_3}{R_1 + R_2 + R_3} \right) + V_{ref} \end{cases} \tag{4.2.17}$$

+24V Voltage monitor の場合は $V_{ref} = 0$ V なので、

$$\begin{cases} V_{in+} = V_{mon} \left(\dfrac{R_2 + R_3}{R_1 + R_2 + R_3} \right) \\ V_{in-} = V_{mon} \left(\dfrac{R_3}{R_1 + R_2 + R_3} \right) \end{cases} \tag{4.2.18}$$

V_{mon} が $V_{TRIPLOW}$ と $V_{TRIPHIGH}$ の場合の V_{in} はそれぞれ under Voltage monitor(下限値)$V_{in+} = 1.167$ (V) であり、over Voltage monitor(上限値)$V_{in-} = 1.228$ (V) であるから

$$\begin{cases} 1.167 = V_{TRIPLOW} \left(\dfrac{R_2 + R_3}{R_1 + R_2 + R_3} \right) \\ 1.228 = V_{TRIPHIGH} \left(\dfrac{R_3}{R_1 + R_2 + R_3} \right) \end{cases} \tag{4.2.19}$$

表 4.5　R_2 の値と $V_{TRIPLOW}$ 及び $V_{TRIPHIGH}$ の関係 (+24 V の場合)

R_2 (kΩ)	$V_{TRIPLOW}$ (V)	$V_{TRIPHIGH}$ (V)
1.2	23.480	25.288
1.5	23.308	25.295
1.8	23.227	25.303
2.2	23.062	25.310
2.26	23.037	25.317
2.7	22.818	25.324

書き直すと

$$\begin{cases} V_{TRIPLOW} = 1.167\left(\dfrac{R_1+R_2+R_3}{R_2+R_3}\right) \\ V_{TRIPHIGH} = 1.228\left(\dfrac{R_1+R_2+R_3}{R_3}\right) \end{cases} \tag{4.2.20}$$

となる。今回は電圧を 24±1 V 前後でモニターするので $V_{TRIPLOW}$ を 23 V、$V_{TRIPHIGH}$ を 25 V と仮定する。$R_1+R_2+R_3$=R_{total} とすると式 (4.2.20) より

$$R_3 = \frac{1.228}{25}R_{total} \approx 0.0491R_{total} \tag{4.2.21}$$

$$R_2 = \frac{1.167}{23}R_{total} - R_3 \approx 0.0016R_{total} \tag{4.2.22}$$

となるので、$R_1 > R_3 > R_2$ と分かる。また、$V_{TRIPLOW}$ が 23 V 付近、$V_{TRIPHIGH}$ が 25 V 付近であるので 2 つの平均は

$$\frac{V_{TRIPLOW}+V_{TRIPHIGH}}{2} = 24\ (\mathrm{V}) \tag{4.2.23}$$

に近い値になる。MAX6458 の入力インピーダンスが大きいので前段の 3 つの抵抗値の合計はできるだけ大きくして電圧降下を防がなくてはならない。そのため $R_1 > R_3 > R_2$ より R_1 を最大値の 1 MΩ として、簡単のために $R_2 = 0\ \Omega$ とすると

$$\frac{V_{TRIPLOW}+V_{TRIPHIGH}}{2} = \frac{1.167\left(\dfrac{R_1+R_3}{R_3}\right)+1.228\left(\dfrac{R_1+R_3}{R_3}\right)}{2} \tag{4.2.24}$$

$$24 = \frac{2.395}{2}\frac{1000+R_3}{R_3} \tag{4.2.25}$$

$$R_3 = 52.5\ (\mathrm{k\Omega}) \tag{4.2.26}$$

となるので、既製品で 52.5 kΩ に最も近い値の抵抗値 51.1 kΩ を R_3 の抵抗値として式 (4.2.20) に代入すると

$$\begin{cases} V_{TRIPLOW} = 1.167\left(\dfrac{1000+51.1}{51.1}\right) = 24.005\ (\mathrm{V}) \\ V_{TRIPHIGH} = 1.228\left(\dfrac{1000+51.1}{51.1}\right) = 25.259\ (\mathrm{V}) \end{cases} \tag{4.2.27}$$

となった。

次に R_2 の抵抗値を大きくしていくと表 4.5 に示す結果が得られた

R_2 は $V_{TRIPLOW}$ が 23 V 付近 $V_{TRIPHIGH}$ が 25 V 付近のもので、なおかつ 2 つの電圧差が最も大きい 2.26 kΩ とした。

これより 3 つの抵抗値は $R_1 = 1$ (MΩ)、$R_2 = 2.2$ (kΩ)、$R_3 = 51.1$ (kΩ) と決定した。

4.2.5.4 -15 V Voltage Monitor の場合

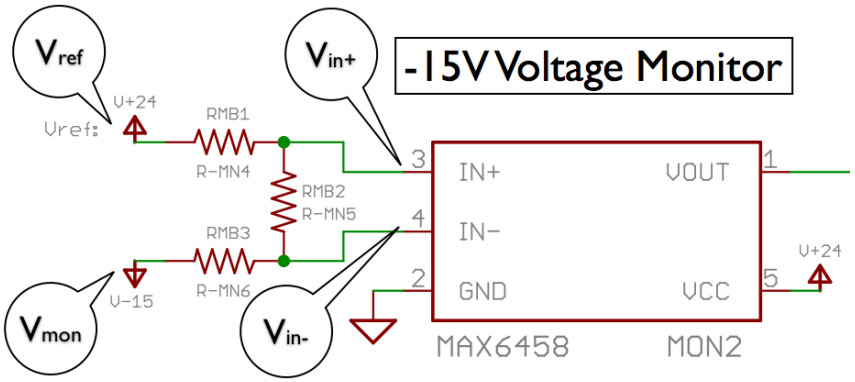


図 4.16　MAX6458 の回路図

考え方は +24 V の場合と同様に考えると、各抵抗値と電圧の関係は以下の式で表される。

$$\begin{cases} V_{in+} = (V_{ref} - V_{mon}) \left(\dfrac{R_2 + R_3}{R_1 + R_2 + R_3} \right) + V_{mon} \\ V_{in-} = (V_{ref} - V_{mon}) \left(\dfrac{R_3}{R_1 + R_2 + R_3} \right) + V_{mon} \end{cases} \tag{4.2.28}$$

$R_{ref} = +24$ V であるので、式 (4.2.28) を書き直すと、

$$\begin{cases} V_{TRIPLOW} = 1.167 \left(\dfrac{R_1 + R_2 + R_3}{R_1} \right) - 24 \left(\dfrac{R_2 + R_3}{R_1} \right) \\ V_{TRIPHIGH} = 1.228 \left(\dfrac{R_1 + R_2 + R_3}{R_1 + R_2} \right) - 24 \left(\dfrac{R_3}{R_1 + R_2} \right) \end{cases} \tag{4.2.29}$$

-15± 1 V でモニターするため $V_{TRIPLOW}$ を-16 V 付近、$V_{TRIPHIGH}$ を-14 V 付近と仮定すると 2 つの平均は

$$\frac{V_{TRIPLOW} + V_{TRIPHIGH}}{2} = -15 \text{ (V)} \tag{4.2.30}$$

に近い値になる。また、$R_2 = 0$ として、R_1 を最大値の 1 MΩ とすると

$$\frac{V_{TRIPLOW} + V_{TRIPHIGH}}{2} = \frac{1.167 \left(\dfrac{R_1 + R_3}{R_1} \right) - 24 \left(\dfrac{R_3}{R_1} \right) + 1.228 \left(\dfrac{R_1 + R_3}{R_1} \right) - 24 \left(\dfrac{R_3}{R_1} \right)}{2} \tag{4.2.31}$$

$$-15 = \frac{2.395}{2} \left(\frac{1000 + R_3}{1000} \right) - 24 \left(\frac{R_3}{1000} \right) \tag{4.2.32}$$

$$R_3 = 710.3 \text{ (kΩ)} \tag{4.2.33}$$

よって既製品でこの値に最も近い 680 kΩ の抵抗を R_3 の抵抗値とした。

この時の R_2 の値ごとの $V_{TRIPLOW}$ 及び $V_{TRIPHIGH}$ の値は表 4.6 になった。

表 4.6　R_2 の値と $V_{TRIPLOW}$ 及び $V_{TRIPHIGH}$ の関係 (-15 V の場合)

R_2 (kΩ)	$V_{TRIPLOW}$ (V)	$V_{TRIPHIGH}$ (V)
0	-14.36	-14.26
10	-14.58	-14.10
30	-15.0	-13.8
62	-15.775	-13.35
68	-15.91	-13.27
75	-16.07	-13.17
100	-16.64	-12.03

$V_{TRIPLOW}$ が-16 V 付近、$V_{TRIPHIGH}$ が-14 V 付近を両方満たす抵抗値の組み合わせは得られなかったので、$V_{TRIPLOW}$ が-16 V 付近となるもので、なおかつ 2 つの電位差が最も小さい 68 kΩ を R_2 の値とした。

これより 3 つの抵抗値は $R_1 = 1$ (MΩ)、$R_2 = 68$ (kΩ)、$R_3 = 680$ (kΩ) と決定した。

4.3 CCD の熱パスの設計

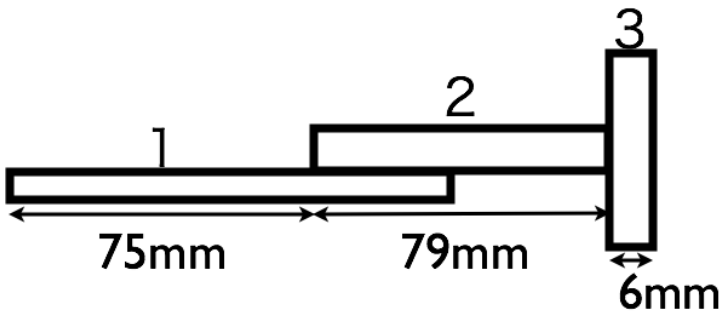


図 4.17 熱パスの略図

CCD のノイズにおいて、出力データに最も影響するものは暗電流である。暗電流は CCD の温度に強く依存し、一般的に 5 から 7 K の温度上昇により暗電流は 2 倍になるため、CCD は低温に冷やす必要があるが、CCD を冷やしすぎると転送効率が低下するため、CCD によって適切な冷却温度は異なる。本研究で使用する CCD の適正冷却温度は約 180 K である。冷凍機の冷却部の温度は約 150 K であるから、デュアー内の CCD と冷凍機を繋ぐ熱パスでの熱伝導による温度上昇は 30 K 以内である必要がある。また、熱パスは金属であるので冷凍機から電流が CCD に流れてしまう可能性があり、これを防ぐために絶縁シートを後に示すパーツ 1 と 2 の間に挟むため、これにより約 5 K 前後の温度上昇が考えられる。

よって熱パスは温度上昇が 20 K 程度であること、及び CCD の中心がデュアー内に開いた窓の中心に位置するように設計した。

以下の計算で設計した熱パスの温度上昇が 20 K 程度になっているかを確かめる。なお、今回の計算では、金属の接続部でのロスや、ネジ穴によるロスは考えていない。

長さ d(m)、断面積 S(m^2)、両端の温度差 ΔT(K) の棒に単位時間に流れる熱量を J(W) とした時の熱伝導率 λ(W/mK) は以下のように表される。

$$\lambda = \frac{Jd}{S\Delta T}\ (\mathrm{W/mK}) \tag{4.3.1}$$

本研究で使用するデュアーから熱放射によって熱パスへ流入する熱量は約 5.1 W であり CCD から熱伝導により流入する熱量は約 3.6 W であることが分かっていため、温度平衡時においての熱パスに流れる熱量 J(W) は

$$J = 5.1 + 3.6 = 8.7\ (\mathrm{W}) \tag{4.3.2}$$

としてよい。

設計した熱パスは図 4.17 のように 3 つのパーツからできているが、図 4.17 のように 1, 2, 3 と番号をつけ 1 と 2 の重なっているところは 2 の長さのみを考慮して、1 は重なっていないところのみを考えて計算した。温度変化 ΔT(K) はタフピッチ銅の熱伝導率 381(W/mk)、長さを d(m)、面積を S(m^2) とすると

$$\Delta T = \frac{8.7 \times d}{0.381 \times S} \tag{4.3.3}$$

となる。

それぞれのパーツでの温度変化を ΔT_1、ΔT_2、ΔT_3、とすると

$$\Delta T_1 = \frac{8.7 \times 75}{0.381 \times 5 \times 26} = 13.17 \text{ (K)} \tag{4.3.4}$$

$$\Delta T_2 = \frac{8.7 \times 79}{0.381 \times 10 \times 26} = 6.94 \text{ (K)} \tag{4.3.5}$$

$$\Delta T_3 = \frac{8.7 \times 6}{0.381 \times 30 \times 30 \times \pi} = 0.05 \text{ (K)} \tag{4.3.6}$$

となる。各パーツのサイズは設計図 (図 4.19 から図 4.22) を参照されたい。

よって、合計の温度変化 ΔT は

$$\Delta T = \Delta T_1 + \Delta T_2 + \Delta T_3 = 20.16 < 30 \text{ (K)} \tag{4.3.7}$$

となり、条件を満たしていることが示されたが、実際は接合部やネジ穴絶縁シートでのロスがあるので $\Delta T > 20$ (K) となり 170 K よりは高温になると考えられる。実際に完成した熱パスの写真を図 4.18 載せる。

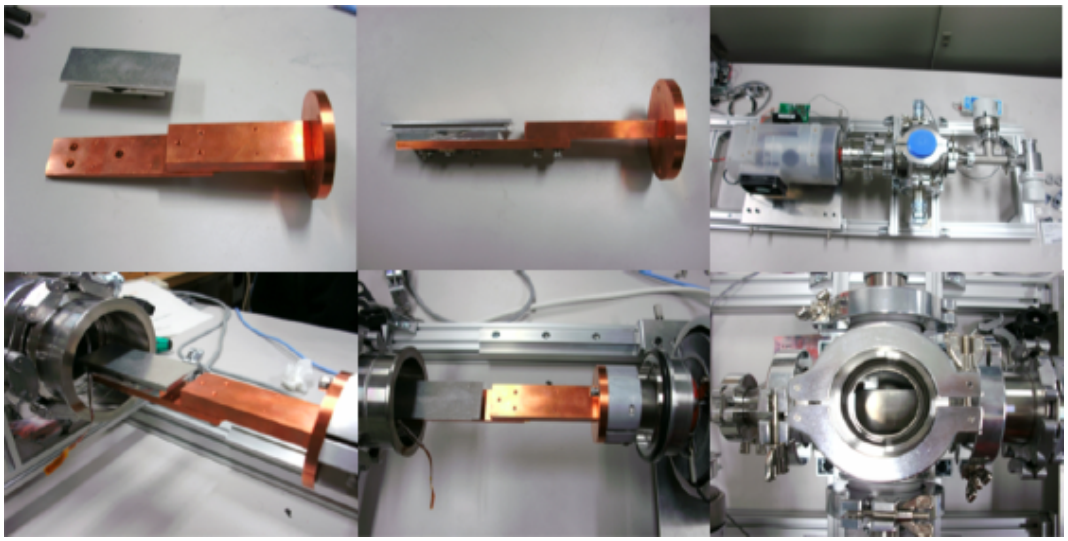


図 4.18　(上段左及び中) 設計した熱パスと 2k×4kCCD 模型。(下段左及び中) 熱パスを冷凍機に取り付けた様子。(上段右) 試験用デュアー。(下段右) デュアーの窓の中心に CCD 模型の中心が入っている様子。

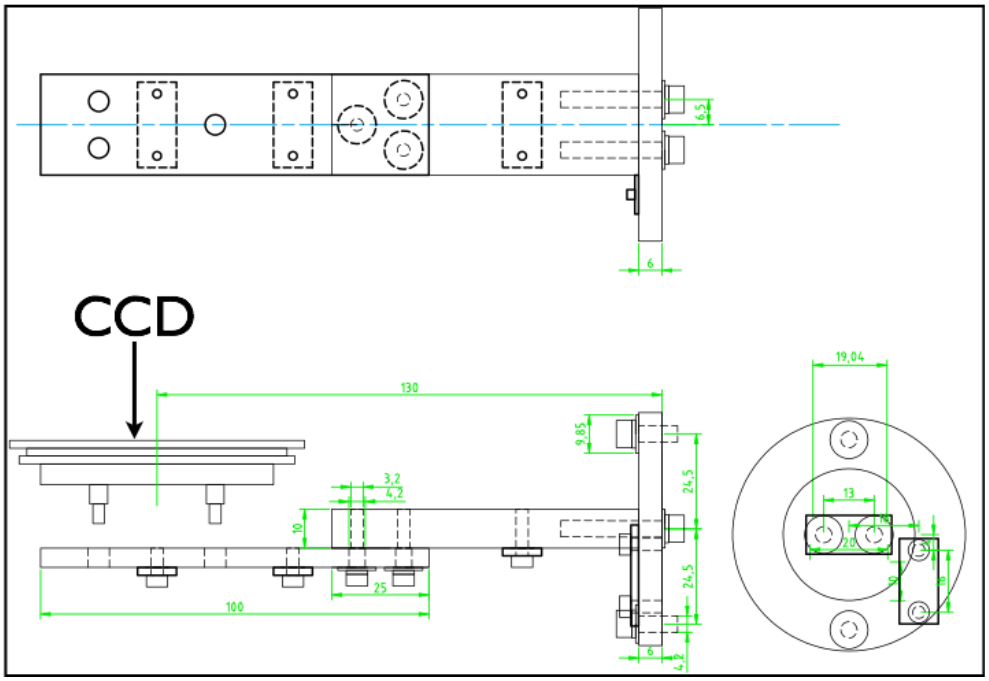


図 4.19　熱パス全体図

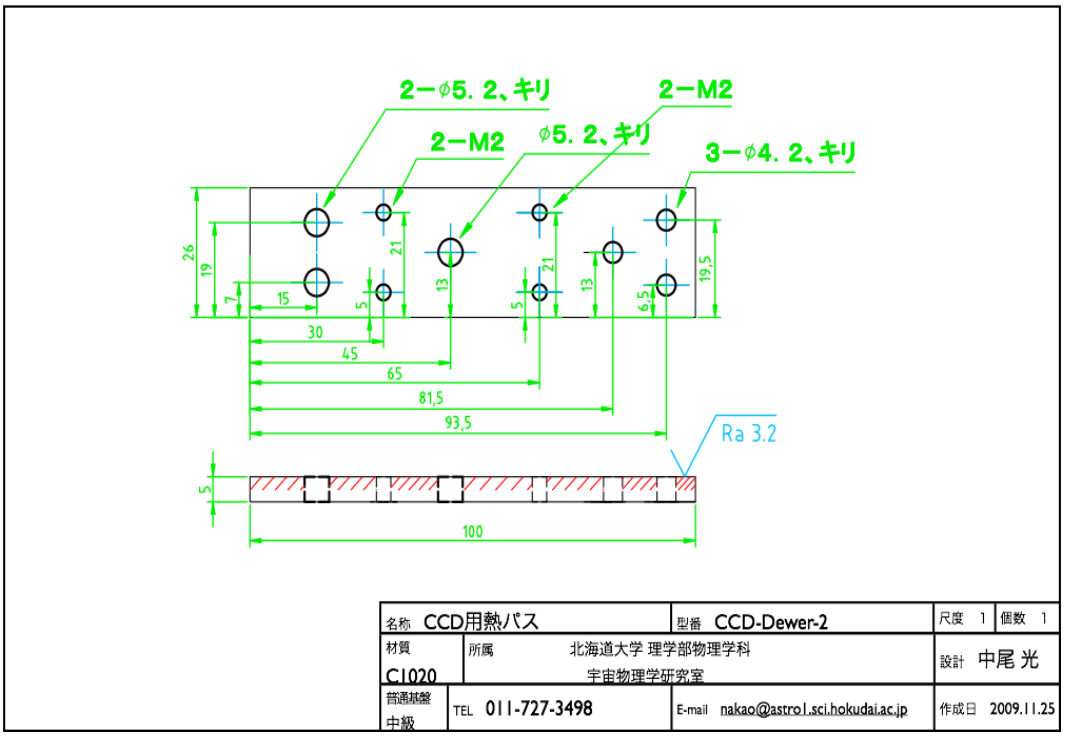


図 4.20　全パーツの概形

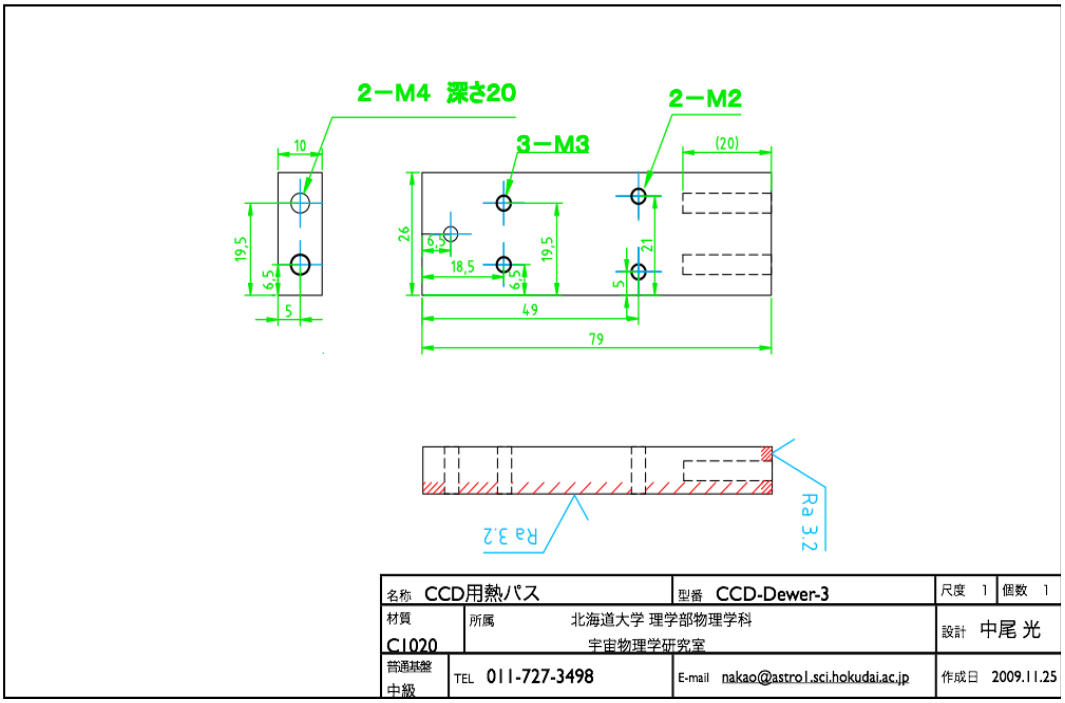

2−M4 深さ20
2−M2
3−M3
10
19.5
6.5
5
26
6.5
18.5
19.5
6.5
(20)
21
5
49
79
Ra 3.2
Ra 3.2
名称 CCD用熱パス
型番 CCD-Dewer-3
尺度 1
個数 1
材質 C1020
所属 北海道大学 理学部物理学科
宇宙物理学研究室
設計 中尾 光
普通基盤 中級
TEL 011-727-3498
E-mail nakao@astro1.sci.hokudai.ac.jp
作成日 2009.11.25

図 4.21

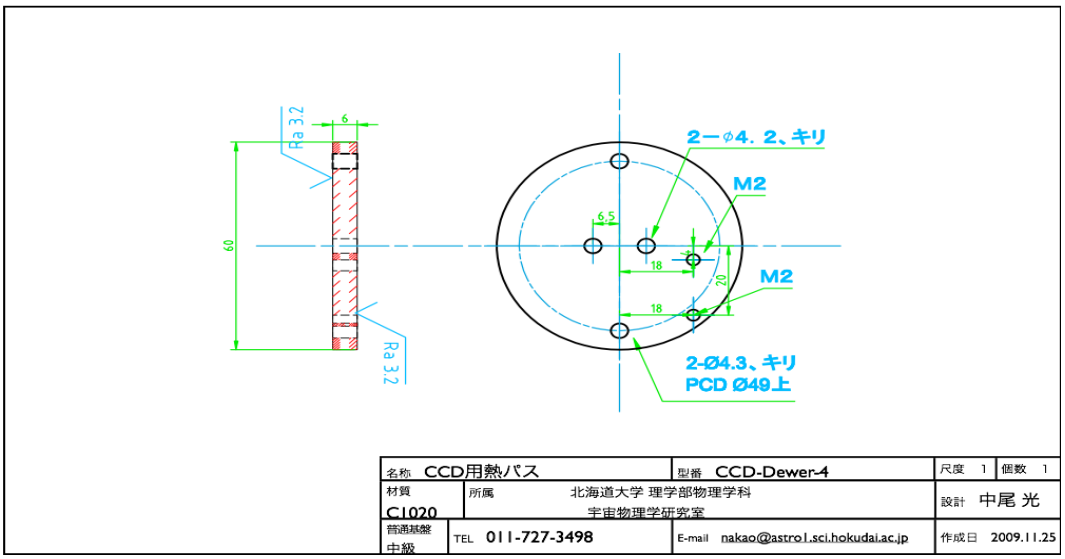

Ra 3.2
6
60
Ra 3.2
2−φ4. 2、キリ
M2
6.5
18
20
M2
18
2-Ø4.3、キリ
PCD Ø49上
名称 CCD用熱パス
型番 CCD-Dewer-4
尺度 1
個数 1
材質 C1020
所属 北海道大学 理学部物理学科
宇宙物理学研究室
設計 中尾 光
普通基盤 中級
TEL 011-727-3498
E-mail nakao@astro1.sci.hokudai.ac.jp
作成日 2009.11.25

図 4.22

4.4　真空計と温度計の制御プログラム

CCD が入るデュアー内は真空、低温の状態に保つ必要がある。そのためリアルタイムでデュアー内の気圧と温度を測定する装置の制御プログラムを製作した。使用した真空計は Pfeiffer Vacuum 社の TPG261、温度計は株式会社ティアンドデイのおんどとり TR-81 である。真空計の基本的な通信フレームは図 4.23 である。図 4.23 内の送信フレームの青文字が命令コマンドであり、＜ CR ＞＜ LF ＞はプログラム上ではヌル文字である。真空計はコマンドを受け取れた場合は応答フレーム＜ ACK ＞を出力する。真空計からの＜ ACK ＞を確認した後で、データ要求フレーム＜ ENQ ＞を送信することで真空計からデータを読み込むことができる。気圧をモニターするプログラムのフローチャートは図 4.24 に示す。

おんどとりの基本的なコマンドフレーム及び応答フレームは 6 バイト固定長である (図 4.26)。その 6 バイトの内 1 バイトがコマンドであり、4 バイトがパラメータ、残りの 1 バイトが SUM である。SUM は図 4.26 にもあるが、コマンドとパラメータと機種コード (0x04 + 0x14) を加算した結果の下位 1 バイトである。またおんどとりは通信の際に、ブレイク信号を送信し 500 msec 待機した後で本体との通信が可能になる。おんどとりの制御プログラムではリアルタイムに指定した時間間隔で温度を取得する他に、おんどとり自身に温度を記録させて後でそのデータをダウンロードするプログラムも作成した。これらのプログラムのフローチャートは図 4.27 から図 4.29 に示す。

作成したプログラムにより試験用デュアー内の空気を真空ポンプで抜いていった時のデュアー内の気圧の変化を記録し図 4.25 を、作成したプログラムにより 2010 年 1 月 3 日の実験室内の気温変化を記録し図 4.30 を得た。本実験ではどちらも指定した時間間隔で両計測器に表示された値が記録できていた。

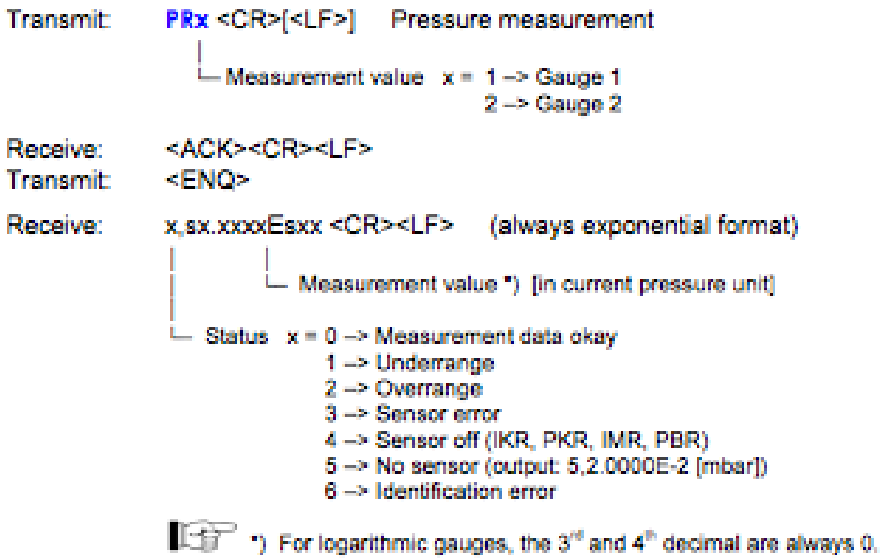


図 4.23　真空計 TPG261 の通信フレーム構成

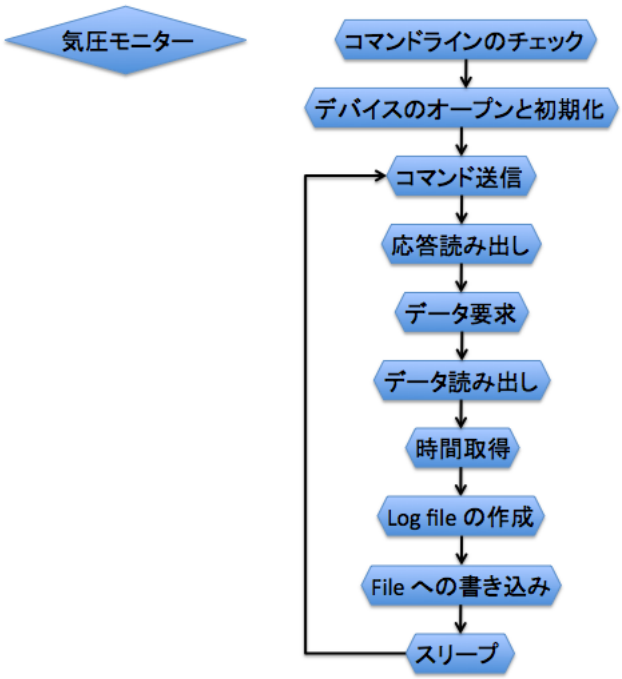


図 4.24　気圧モニターのプログラムフローチャート

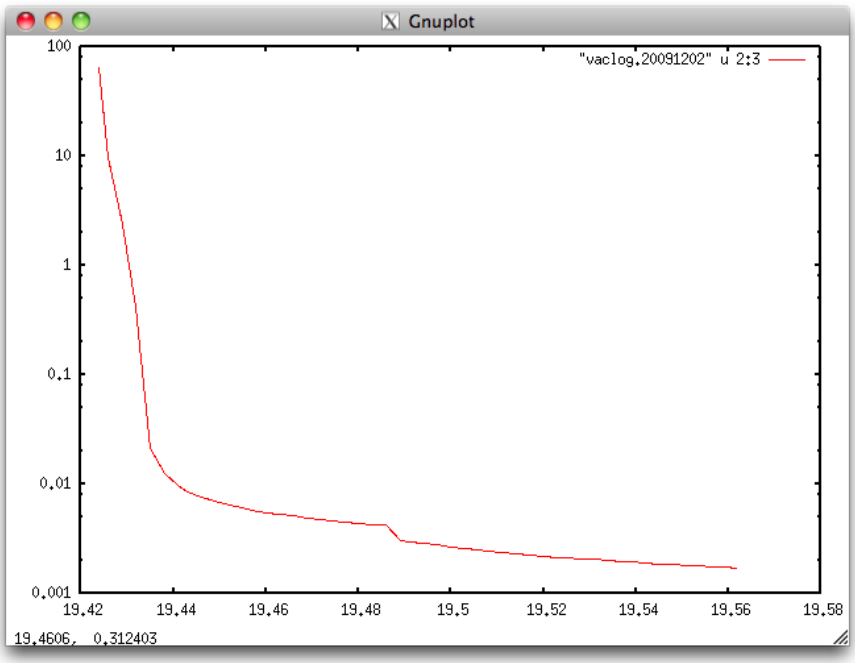


図 4.25　試験用デュアー内の空気を抜いていった時の気圧の変化。縦軸は気圧 (Pascal)、横軸は時刻 (hour)。

コマンドフレーム構成

先頭					末尾
コマンド	パラメータ1	パラメータ2	パラメータ3	パラメータ4	SUM

SUM　　コマンドとパラメータと機種コード(0x04 + 0x14)
を加算した結果の下位1バイト

応答フレーム構成

先頭					末尾
コマンド	パラメータ1	パラメータ2	パラメータ3	パラメータ4	SUM

図 4.26　おんどとりの通信フレーム構成

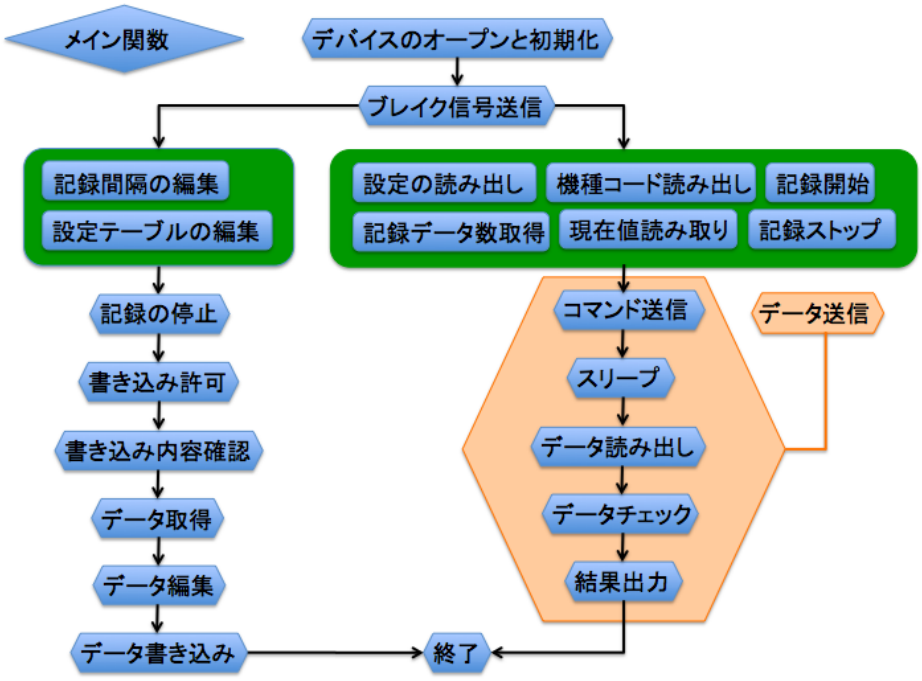


図 4.27　メイン関数のフローチャート

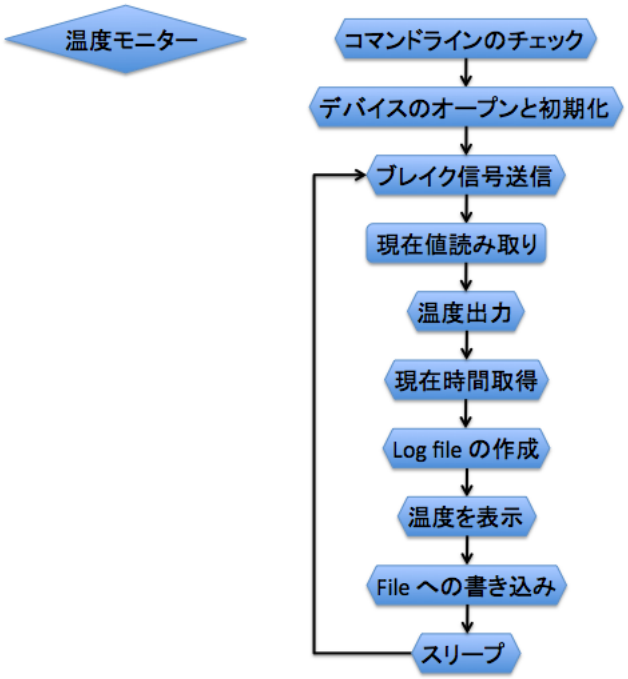


図 4.28　温度モニターのプログラムフローチャート

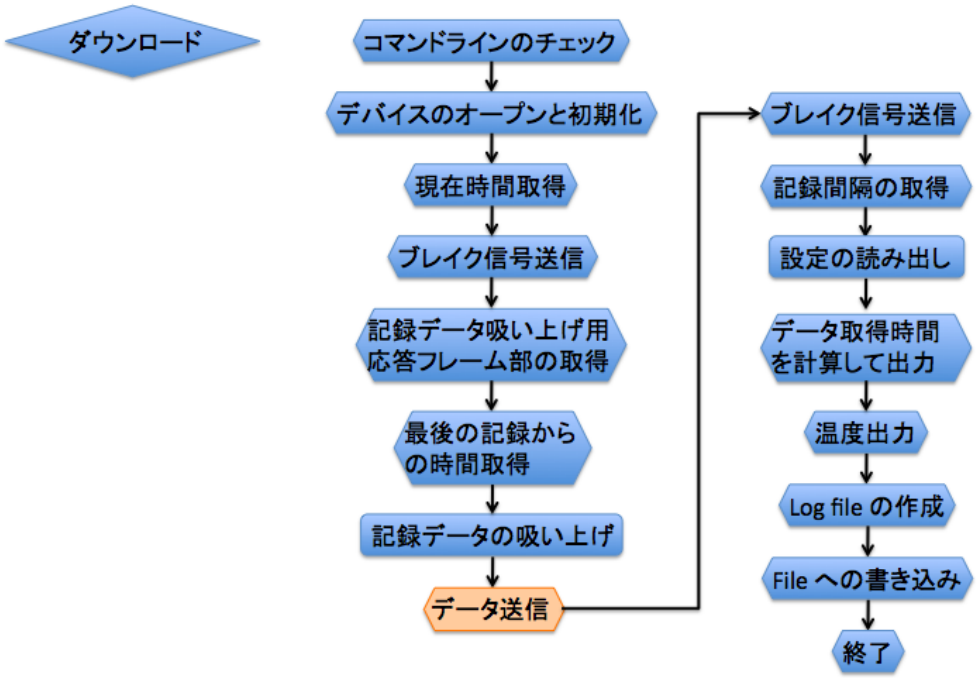


図 4.29　データダウンロードのプログラムフローチャート

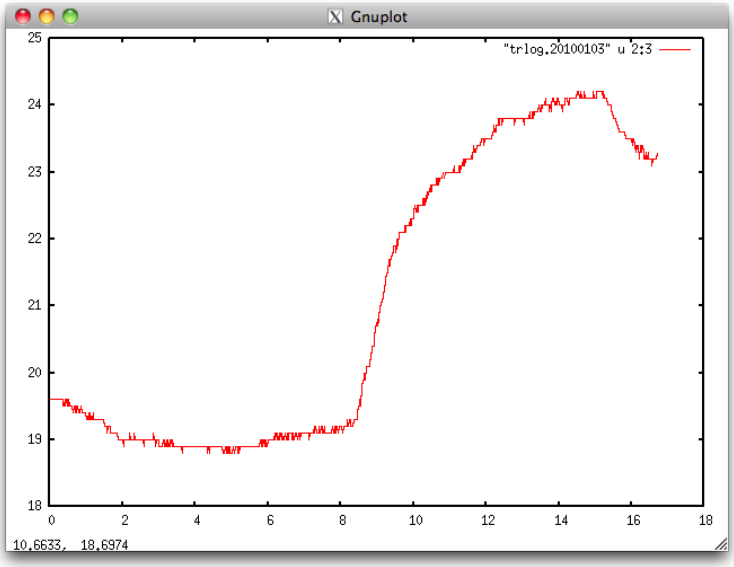


図 4.30　2010 年 1 月 3 日の実験室内の気温変化。縦軸は温度 (℃)、横軸は時刻 (hour)。

5　まとめ

本研究では東京大学大学院理学系研究科天文学教育研究センター木曽観測所の 105 cm シュミット望遠鏡を用いて近傍の銀河団 A2634 及び A426 を B、V、R の広帯域フィルター及び、Ha6737、Ha6577、N688 の狭帯域フィルターを用いて観測を行った。天候により十分な観測データは得られなかったが、A2634 を Ha6737 フィルター観測した結果において、図 2.7 に矢印で示した銀河が上下に広がった構造が確認できた。この広がった構造は Hα 輝線で観測されているため、その銀河が渦巻銀河である可能性があるが、Dressler et al (1980) はその銀河をレンズ状銀河 (S0) に分類していた。この食違いについても、露光時間を増やすことでより詳細な議論ができると思われる。

さらに研究を進めるため名寄市に建設されている 1.6m 光学赤外線望遠鏡用に分光撮像装置を製作しており、私は CCD の読み出し回路の製作を行っている。読み出し回路の 4 つのボードのうち、IF ボードは試験も終了しほぼ完成している。またその他の 3 枚のボードも試作品の設計はできており発注済みである。

読み出し回路以外にも熱パスの設計や真空計及び温度計の制御プログラムの作成等も行った。

修士課程ではこの分光撮像装置を完成させて、本観測と同様の撮像観測と分光による観測を進めていく予定である。

謝辞

本研究に協力していただいた多くの方々に心より御礼申し上げます。指導教官の徂徠先生には私が学部４年生であるにもかかわらず、木曽での観測や約２ヶ月の東京での作業など多くの経験をさせていただき、本研究でも多くの御指導をいただきました。また、研究室の藤本先生、羽部先生、南谷先生には講義等でご指導いただきました。

木曽観測所では所員の青木さん、征矢野さん、猿楽さんのご協力により観測する事ができました。また、観測所に滞在中、観測所の皆さんにも大変お世話になりました。

東京大学に滞在中は、土居先生、宮田先生、酒向先生、学生の加藤さん、実験室の皆さんに大変親切にしていただきました。特に何も知らない私に読み出し回路製作のことを丁寧に御指導してくださった酒向先生、加藤さんに深く感謝します。

この１年を先輩達に囲まれて楽しく過ごせたこと、そして非常に研究熱心な同期と過ごせたことに感謝します。

Reference

[1] Hubble, E. : The Reallm of the Nebulae, (New Haven : Yale Univ. Press). (1936)

[2] Dressler et al : A CATALOG OF MORPHOLOGICAL TYPES IN 55 RICH CLUSTERS OF GALAXIES, (Apjs,42:565). (1980)

[3] C R KITCHIN : ASTROPHYSICAL TECHNIQUES, University of Hertfordshire Observatory (1998)

[4] Donald E. Osterbrock, Gary J Ferland : Astrophysics of Gaseous Nebulae and Active Galactic Nuclei, University Science Books (2006)

[5] 家　正則編：現代の天文学　宇宙の観測Ⅰ ー光・赤外天文学, 日本評論社 (2007)

[6] 岡村　定矩著：銀河系と銀河宇宙, 東京大学出版会 (1999)

[7] 吉永　淳著：アナログ回路, アルテ (1998)

[8] 奥澤　熙著：はじめて見るオペアンプ, 誠文堂新光社 (2005)

以下マニュアル

[9] DS90UR241/DS90UR124　データシート, National Semiconductor,

`http://www.national.com/JPN/ds/DS/DS90UR124.pdf`

[10] FPGA　データシート, XILINX,

`http://www.xilinx.com/support/documentation/data_sheets/ds529.pdf`

[11] MAX6458　データシート, MAXIM,

`http://datasheets.maxim-ic.com/en/ds/MAX6457-MAX6460.pdf`

[12] PEX-292144　データシート, interface,

`http://www.interface.co.jp/download/manual/meu/meuj292144_11_1.pdf`

[13] TR-81　通信プロトコル, ティアンドデイ,

[14] TPG261　通信プロトコル, Pfeiffer Vacuum,

`http://www.sls.psi.ch/controls/hardware/pdf/pfeiffer-TPG-262.pdf`

6　Appendix

東京大学に滞在している間、読み出し回路の開発以外にも分光撮像装置の開発に必要となる作業を数多く手伝わせていただけた。以下ではその作業を簡単に記している。

温度コントローラー実験

読み出し回路の性能は、外気温にある程度影響されると考えられるため、様々な気温で読み出し回路の性能を評価する必要がある。今回は、発泡スチロール容器と温度センサー、ヒーターを用いて、発泡スチロール容器内の温度を室温以上に維持できるかを試験した。各装置の配線図は図 6.1 である。

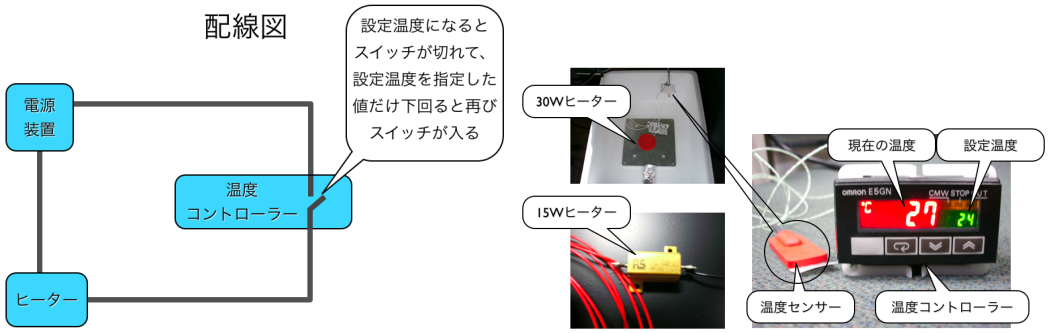


図 6.1　配線図 (左)、装置の写真 (右)

本実験では室温が 290 K 前後の時に 15 W のヒーターで容器内の温度を 295 K に保てるかを試験した。温度コントローラーの温度計は感度が鈍いため、温度とりで同時により正確な温度を記録し、結果は図 6.2 のようになり設定温度から ±1 K 前後で維持できることが分かった。

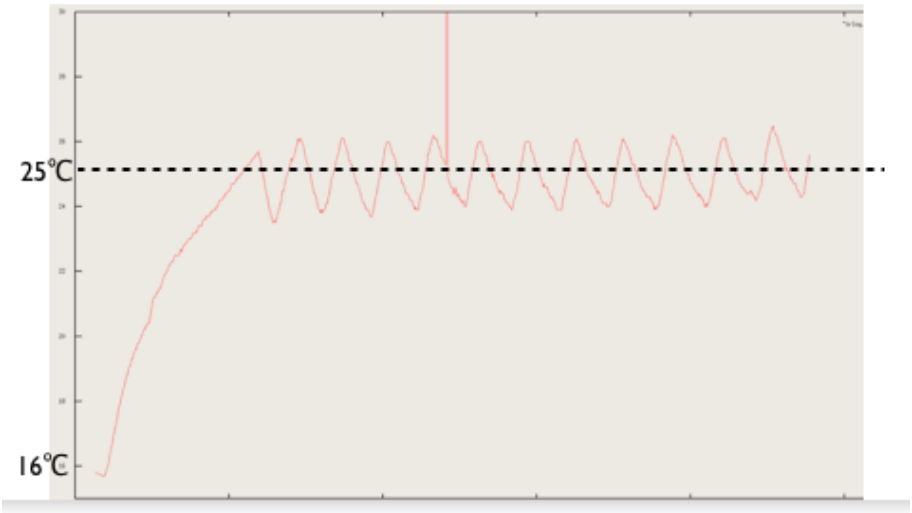


図 6.2　温度コントロール実験結果

ハーネスの設計

DRV ボードと CCD を繋ぐケーブルは芯数が 26 本でノイズを最小限に抑えられるものが望ましい。市販品ではこの条件を満たすハーネスが無かったので特注した。コネクタは DRV ボードの大きさに合っており、かつ芯数が 26 本の高密度 Dsub コネクタ (HD-26SP オス) を選んだ。またコネクタを覆うフードは EMI(Electromagnetic Interference、電波障害) 対策がされたもの (J-C15-2C) を、ケーブルは最もノイズを軽減すると思われる 3 重シールド信号用ケーブル (NASW-25-13P) を選んだ。

実際の発注した際の図面は図 6.3 及び図 6.4 である。図 6.4 はケーブルの結線図であり、ツイストペア (ノイズ軽減のため 2 本の芯線をツイストして 1 対としている) の指定をしている。

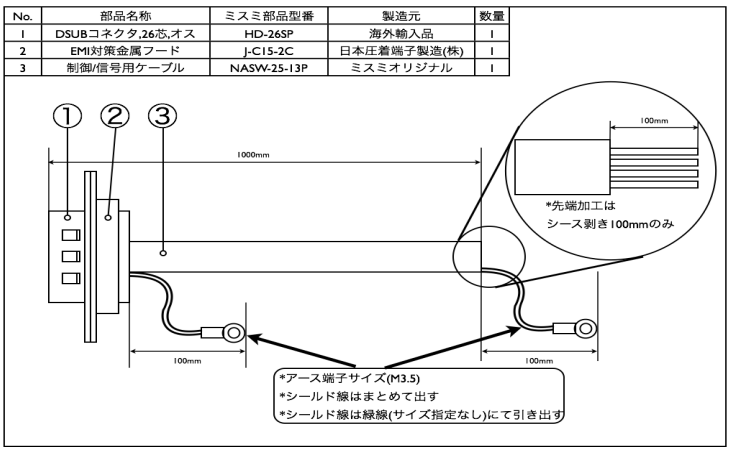


図 6.3　ハーネスの図面

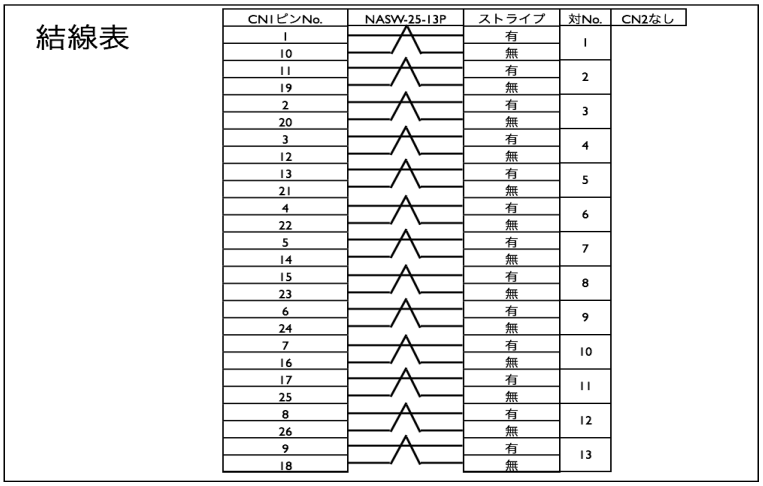
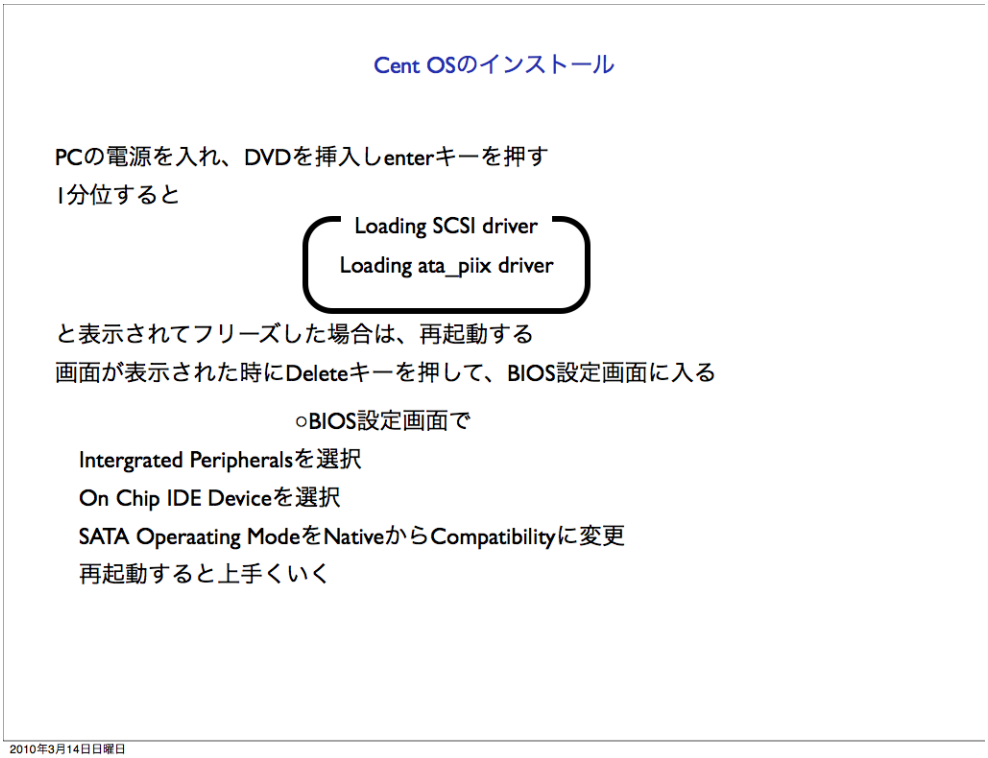


図 6.4　ハーネスの結線図

CCD コントローラーのユーザーズマニュアル

東京大学滞在中に読み出し回路を制御する PC のセットアップ方法、及び正しくデータ通信ができているかをテストするサンプルクロックの使い方と作り方をまとめ、読み出し回路のユーザーズマニュアルを作成した。以下にそれを載せる。



Cent OSのインストール

○インストール画面の表示と選択

nextをエンター

What languate would.... → Japaneseを選択

キーボードを選択 → 日本語

パーティションの選択 → 全て削除してデフォルトにする

ハードドライブ	マウントポイント	タイプ	フォーマットする	容量(MB)	開始	終了
/dev/hda1	/boot	ext3	○	101	1	13
/dev/hda2	VolGroup01	LVMPV	○	238370	14	30401

確認して → 次へ

GRUBブートローダは/dev/hda上にインストールされます → デフォルトをチェックして 次へ

ネットワークデバイス → DHCP経由

地域 → 東京

rootのpassword → ○○○○○○

2010年3月14日日曜日

Cent OSのインストール

ソフトウェア → Desktop-Gnomeと今すぐカスタマイズするをチェックする

デフォルトから追加したもの

アプリケーション	全てチェック
開発	全てチェック
サーバー	サーバー設定ツール
ベースシステム	全てチェック

次へをクリック

次へをクリック

ファイルのフォーマット中… → 30分位待機

インストール → 60分位待機

DVDを取り出して再起動

2010年3月14日日曜日

セットアップ

ようこその画面 → 進むをクリック

ファイアーウォール → 有効でSSHのみ信用

SELinux → SELinux設定　Enforcingのまま

Kdump → Kdumpを有効

合計システムメモリー	2017MB
Kdumpメモリー	128MB
使用可能メモリ	1889MB

後で再起動しますか？ → yes

日付と時刻 → 設定した。ネットワークプロトコルは無効のまま

ユーザーの作成 → ユーザー名、フルネームをccdで作成

→ ネットワークログインを使用するは無視

サウンドカード → 再生したが何も聞こえなかった

→ 聞こえないを選択

→ エラーが表示されてOKをクリック

→ 何も変更せずに進む

2010年3月14日日曜日

セットアップ

追加のCD → 終了

再起動します → OK

再起動中にKdumpにエラーが出たが問題なく起動出来た

2010年3月14日日曜日

DIOボードのPCへの設置

PCの電源を落とす

2枚のDIOボードのロータリースイッチ(RSW1)の値は1と2に設定した

(ロータリースイッチの値はそれぞれ異なる値に設定する事)

DIOボードを2枚PCに取り付けた

電源を入れてlogin

2010年3月14日日曜日

DIOボードのPCへの設置(インストール)

http://www.interface.co.jp/download/search.aspでユーザーID登録

今回は法人登録、北海道大学、中尾で登録した

(ユーザーID : L7100ZA01)(Password : ccd44d)

同じサイト内でGPG-2X7CCを検索しダウンロード

(ダウンロードしたディレクトリは /tmp)

スーパーユーザーになって、アーカイブを解凍

```
%su
#cd /tmp
#tar xvzf gpg2x72c_i386_016113.tgz
```

インストーラーを起動し、指示に従ってインストール

#sh install (インストールに日本語を使うと文字化けしたので英語を選択)

y

1 ← Lnux driverの1を選択

y ← ここにインストールするか？

t ← ノーマルで良いか？

y ← インストールするか？

n ← readmeを読むか？

インストール完了

2010年3月14日日曜日

ドライバモジュールの組み込み

スーパーユーザーになる

```
%su
```

ドライバ組み込み用シェルスクリプトを実行し、ドライバモジュールの組み込みを行う

```
#cd /usr/src/interface/gpg2x72c/i386/linux/drivers
#sh insdiobm.sh
```

このコマンドはパスが通っていないために実行出来なかったのでパスを通す

同じディレクトリで(/usr/src/interface/gpg2x72c/i386/linux/drivers)

```
#vi insdiobm.sh
```

3行目の uname の前に /bin/ を挿入

5行目の modprobe の前に /sbin/ を挿入

7,8行目の insmod の前に /sbin/ を挿入

```
:wq
```

で編集完了

もう1度シェルスクリプトを実行する

```
#sh insdiobm.sh
```

デバイス番号設定ユーティリティを実行し、デバイスノードを作成する

```
#sh setup.sh
```

ドライバソフトウェア名を聞かれる → 2x72c と入力

デバイス番号が表示される → 99 と入力し終了

2010年3月14日日曜日

カーネルバージョン

スーパーユーザーになる

```
%su
```

(インストールディレクトリ/common)ディレクトリにある、全てのモジュールに対してコンパイルを行いインストールする

(dpg0100用)

```
#cd /usr/src/inerface/common/dpg0100/src
#make
#make install
```

(dpg0101用)

```
#cd /usr/src/inerface/common/dpg0101/src
#make
#make install
```

linuxのドライバのsrcディレクトリに移動し、モジュールのコンパイルを行いインストールする

```
#cd /usr/src/inerface/gpg2x72c/i386/linux/drivers/src
#make
#make install
```

各ドライバに用意されているドライバ組み込み用シェルスクリプトを同様に実行する

```
#cd ../
#sh insdiobm.sh
```

2010年3月14日日曜日

デバイス番号の確認

PEX-292144を2台インストールした時のデバイス番号設定ユーティリティの出力

```
%su
#sh setup.sh
```

ドライバソフトウェア名を聞かれるので

2x72c と入力

右図が表示される

99と入力して終了

```
=============================================================
Ref.ID | Model                          | RSW1 |  Type | Device No.
-------------------------------------------------------------
     1 | PEX-292144                     |    1 |    DI |    19
-------------------------------------------------------------
     2 | PEX-292144                     |    1 |    DO |    20
-------------------------------------------------------------
     3 | PEX-292144                     |    2 |    DI |    21
-------------------------------------------------------------
     4 | PEX-292144                     |    2 |    DO |    22
=============================================================

************** Command ***************
  1. Change the device number.
  2. Delete the device number.
  3. Load new device setting file.
  4. Run the initialization program.
  5. Run the CardBus ID setup utility.
 99. Exit the program.
**************************************
```

入出力ボードにより、デバイス番号が異なる

Type = DI　：入力用ポート

Type = DO ：出力用ポート

clpには出力用ポートを、frpには入力用ポートを使用する

2010年3月14日日曜日

ユーザーccd内にkwfcのディレクトリを作成する

ccdのディレクリに移動してkwfcのディレクトリを作成する

```
%cd /home/ccd
%mkdir kwfc
```

wikiからkwfc_V01_20100122.tgzをダウンロード

ダウンロード先のディレクトリで

```
%mv kwfc_V01_20100122.tgz /home/ccd/kwfc
%cd /home/ccd/kwfc
%zxvf kwfc_V01_20100122.tgz
```

これでkwfcのディレクトリにkwfcが展開される

2010年3月14日日曜日

ds9のインストール

http://hea-www.harvard.edu/RD/ds9/　にアクセス

Linux (DS9 Version6.0 Binarles)をクリックして保存する

保存先のディレクトリに移動して展開する

```
%cd 保存先のディレクトリ
%tar xvfz ds9.linux.6.0.tar.gz
```

スーパーユーザーになってds9を移動させる

```
%su
#mv ds9 /usr/bin
#exit
```

ds9を起動する時はターミナル上で

```
%ds9 &
```

で起動する

2010年3月14日日曜日

サンプルクロックソースファイルnakao.srcをfits画像にしてds9で表示してみる

端末①を開きclpのディレクトリに移動する

```
%cd /home/ccd/kwfc/kwfc/kac/clp
```

clp内のソースファイル(.src)をバイナリファイル(.bin)に変換する

```
%./cpc data/nakao.src
```

(nakao.srcのデータ数が分からない場合は次のコマンドで確認しておく

```
%./bitview data/nakao.bin
```

により16進数で表示されるので10進数に直す)

別の端末②を開きfrpのディレクトリに移動する

```
%cd /home/ccd/kwfc/kwfc/kac/frp/
```

binデータを受け取りdatにするためのfrpコマンドを実行

```
%./frp 21 67108864 data/nakao.dat
```

端末①でbinファイルを出力するclpコマンドを実行

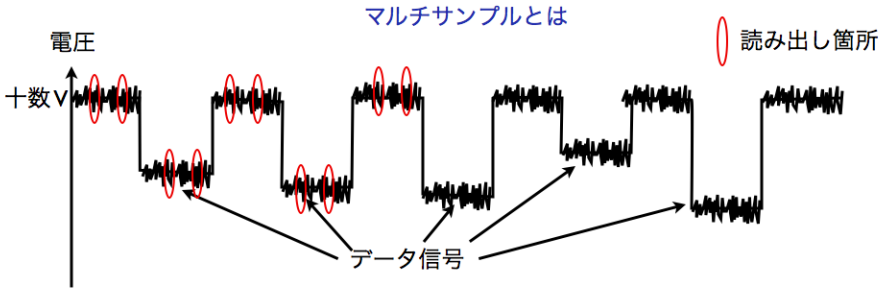
```
%./clp 20 data/nakao.bin
```

r2fのディレクトリでdatをfitsに変換し、ds9で確認する

```
%cd ../../fits
%r2f 0 ../kac/frp/data/nakao.dat data/nakao.fits
%ds9 & data/nakao.fits
```

2010年3月14日日曜日

マルチサンプルとは



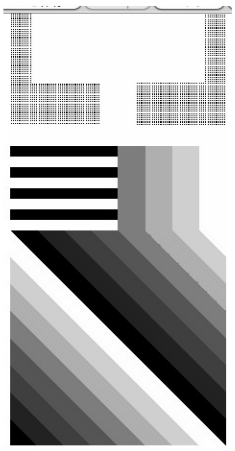
CCDからのデータ信号は図の用に十数Vの基準電圧の中に数 mV出力される。
観測されたデータは基準電圧とデータ信号の電圧の差であるので、基準電圧とデータ信号の電圧の両方を読み出す必要がある。
この時各信号にはノイズが混ざっているため、1つの信号を何度かサンプルして平均を取ることでノイズの影響を取り除く事が出来る。
1つの信号のサンプル回数がマルチサンプル数である。
マルチサンプル数を多くすると、データの質が上がるが読み出し時間は倍増する。

2010年3月14日日曜日

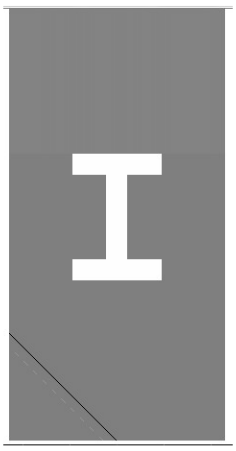
r2fコマンドのテスト

nakao.srcはr2fのマルチサンプル数を0から4まで変える事で異なるfits画像になります

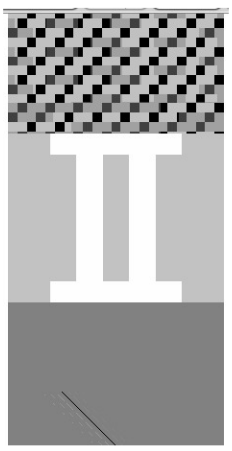
マルチサンプル数０



マルチサンプル数１



マルチサンプル数２

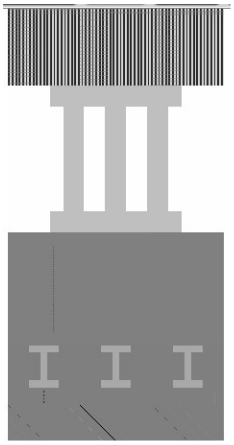


2010年3月14日日曜日

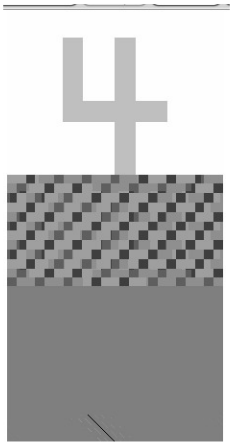
r2fコマンドのテスト

nakao.srcはr2fのマルチサンプル数を0から4まで変える事で異なるfits画像になります

マルチサンプル数３



マルチサンプル数４

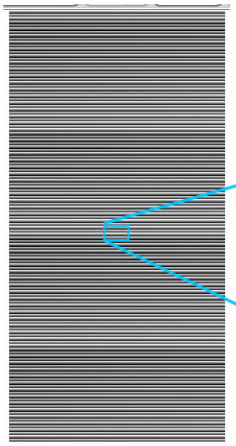


2010年3月14日日曜日

その他のソースファイルとfits画像

ソースファイルs2k4k.srcをfis画像にすると以下の画像が得られます

マルチサンプル数０

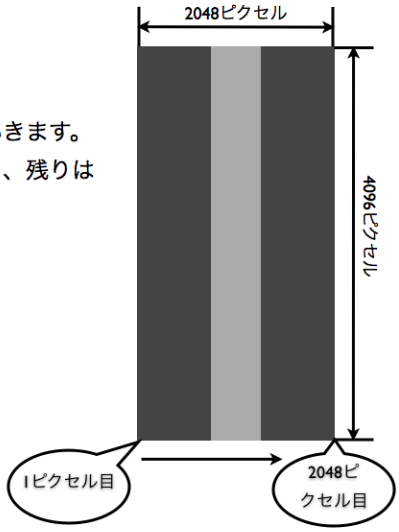


拡大図

2010年3月14日日曜日

簡単なクロックソースファイルを作ってみる

方針
右図の様なfits画像になるソースファイルを作ります
ソースファイルの情報はfits画像の左下から右方向へ
出力され、2048ピクセルごとに上の行に出力されていきます。
なので、まずは底辺となる2048ピクセル目までを作り、残りは
同じ操作の繰り返しで作ります。



2010年3月14日日曜日

簡単なクロックソースファイルを作ってみる

構成
ソースファイルは大きくParameter、Definition、
Command Data、Bit Dataの4つの構成です
○Parameterには使用する整数値を記述します
○Definitionでは使用する変数を定義します
○Command Dataでコマンドを組み合わせて出力す
るデータを作ります
○Bit Dataではデータの元となるビットをまとめた
ビットパケットを作ります

#の後は行末までコメントアウトされます
まずは、構成の枠だけを書いてみます

ファイル名は拡張子が .src であれば何でも良いの
でsample.srcとしました

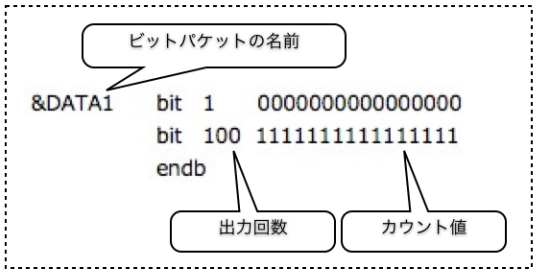
sample.src

```
1  #-------------------------------------------#
2  #                                           #
3  #          sample source file               #
4  #                                           #
5  #                2010/02/10 H.Nakao         #
6  #-------------------------------------------#
7
8  #---------------------- Parameters ----------------------#
9
10
11 #---------------------- Definition ----------------------#
12
13
14 #---------------------- Command Data --------------------#
15
16
17 #---------------------- Bit Data ------------------------#
```

2010年3月14日日曜日

簡単なクロックソースファイルを作ってみる

まずは、fits画像のピクセルのカウント数に対応するBit Dataの編集を行います
Bit Data内のビットパケットの構成は下図のように&で始まりendbで終わります



上図の場合、&DATA1を1回実行するとカウント数0のビットが1個とカウント数65535のビットが100個出力されます

sample.src

```
#-------------------------------------------#
#                                           #
#          sample source file               #
#                                           #
#                2010/02/10 H.Nakao         #
#-------------------------------------------#

#---------------------- Parameters ----------------------#


#---------------------- Definition ----------------------#


#---------------------- Command Data --------------------#


#---------------------- Bit Data ------------------------#
```

2010年3月14日日曜日

簡単なクロックソースファイルを作ってみる

今回は2種類のビットパケットがあれば目的のfitsが出来るのでビットパケットは2つ作ります

右図のビットパケットを&DATA1、&DATA2、&DATA1順番に実行すると、合計のビット数が2048になって、fits画像の底辺が出来上がります

Bit Dataの最初には必ず
START_BIT_DATA と入力します

sample.src

```
#-------------------------------------------#
#                                           #
#          sample source file               #
#                                           #
#                2010/02/10 H.Nakao         #
#-------------------------------------------#

#---------------------- Parameters ----------------------#


#---------------------- Definition ----------------------#


#---------------------- Command Data --------------------#


#---------------------- Bit Data ------------------------#

START_BIT_DATA

&DATA1          bit  774     1111111111111111
                endb

&DATA2          bit  500     0000000000000000
                endb
```

2010年3月14日日曜日

簡単なクロックソースファイルを作ってみる

次はBit Dataに作ったビットパケットをCommand Dataで出力出来るようにします

ビットパケットを出力するコマンドは
outp です

この様に記述すると&DATA1で指定したビットパケットを出力出来ます

&DATA1、&DATA2、&DATA1の順番で出力するには右図のように書きます
Command Dataの最後には必ずhaltを書きます

sample.src

```
#-------------------------------------------#
#                                           #
#          sample source file               #
#                                           #
#                2010/02/10 H.Nakao         #
#-------------------------------------------#

#---------------------- Parameters ----------------------#


#---------------------- Definition ----------------------#


#---------------------- Command Data ----------------#

                outp &DATA1
                outp &DATA2
                outp &DATA1
                halt

#---------------------- Bit Data ----------------------#

START_BIT_DATA

&DATA1          bit   774     1111111111111111
                endb

&DATA2          bit  500      0000000000000000
                endb
```


簡単なクロックソースファイルを作ってみる

ここまでで、fits画像の1列目は出来たので16行目から18行目のコマンドを4096回繰り返すことで、fits画像を完成させます

指定した回数を繰り返すコマンドは、
1つの定数と変数、コマンドloadとcjmp を組み合わせて作ります
まずは、使用する定数と変数を定義します
定義する時には $名前 整数 のように書き、
今回の定数はループする回数なので$NLP1
変数は$LREG1としました

定義すると右図のようになります
$NLP1の整数はループ回数
$LREG1の整数は通し番号の様なものです。他の変数を定義した時は異なる整数を使用してください

sample.src

```
#-------------------------------------------#
#                                           #
#          sample source file               #
#                                           #
#                2010/02/10 H.Nakao         #
#-------------------------------------------#

#---------------------- Parameters ----------------------#

$NLP1           4096

#---------------------- Definition ----------------------#

$LREG1          1

#---------------------- Command Data ----------------#

                outp &DATA1
                outp &DATA2
                outp &DATA1
                halt

#---------------------- Bit Data ----------------------#

START_BIT_DATA

&DATA1          bit   774     1111111111111111
                endb

&DATA2          bit  500      0000000000000000
                endb
```

簡単なクロックソースファイルを作ってみる

コマンド load と cjmp は右図のように組み合わせて使います

load $LREG1 $NLP1 は、$LREG1に$NLP1の値4096を代入します

cjmp $LREG1 @ROOP1 は、$LREG1の値が1以上の場合に$LREG1の値を-1して@ROOP1に飛びます

右図のように記述すると、コマンド①を4096回繰り返す事が出来ます

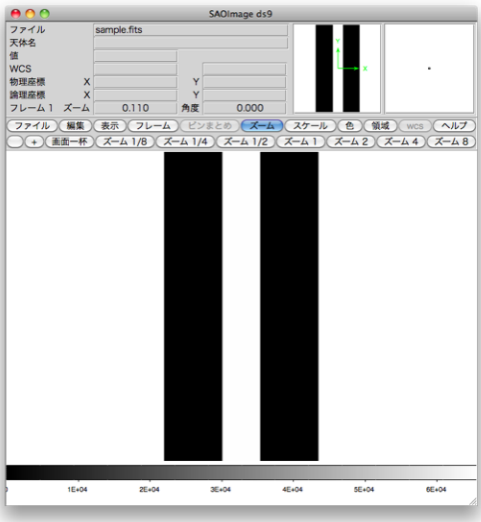
sample.src

```
1  #-------------------------------------------#
2  #                                           #
3  #          sample source file               #
4  #                                           #
5  #            2010/02/10 H.Nakao             #
6  #-------------------------------------------#
7
8  #---------------------- Parameters ----------------------#
9
10 $NLP1          4096
11
12 #---------------------- Definition ----------------------#
13
14 $LREG1         1
15
16 #---------------------- Command Data ----------------#
17
18                 load $LREG1 $NLP1
19 @ROOP1          outp &DATA1
20                 outp &DATA2        <-- ①
21                 outp &DATA1
22                 cjmp $LREG1 @ROOP1
23                 halt
24
25 #---------------------- Bit Data ----------------------#
26
27 START_BIT_DATA
28
29 &DATA1          bit  774    1111111111111111
30                 endb
31
32 &DATA2          bit  500    0000000000000000
33                 endb
34
```

2010年3月14日日曜日

簡単なクロックソースファイルを作ってみる

完成したソースファイルとそのファイルをfitsに変換してds9で表示したものです



sample.src

```
1  #-------------------------------------------#
2  #                                           #
3  #          sample source file               #
4  #                                           #
5  #            2010/02/10 H.Nakao             #
6  #-------------------------------------------#
7
8  #---------------------- Parameters ----------------------#
9
10 $NLP1          4096
11
12 #---------------------- Definition ----------------------#
13
14 $LREG1         1
15
16 #---------------------- Command Data ----------------#
17
18                 load $LREG1 $NLP1
19 @ROOP1          outp &DATA1
20                 outp &DATA2
21                 outp &DATA1
22                 cjmp $LREG1 @ROOP1
23                 halt
24
25 #---------------------- Bit Data ----------------------#
26
27 START_BIT_DATA
28
29 &DATA1          bit  774    1111111111111111
30                 endb
31
32 &DATA2          bit  500    0000000000000000
33                 endb
34
```

2010年3月14日日曜日