

目次・まえがき

目次

目次・まえがき		1
部員自己紹介		2~3
ロケット	yone	4~11
学校から配布された端末でゲームをする話	ちーしゃ	12~19
遺伝的アルゴリズム	そすいー	20~22
ラジオの仕組みと自作の方針	T. R.	23~25
Live2D でキャラクターを動かす！	T. R.	26~31
四輪オムニの制御と制作秘話	K. T.	32~35
美少女の髪の毛作ってみた	K. T.	36~40
適当すぎる Blender 使い方講座	よつばのくろおばあ	41~45
帝国海軍の電波発信機	眠い人	46~50
ARDF について	K. K.	51~53
トランジスタで UART を作ろう	きゅうた	54~76
あとがき・その他		77

まえがき

本日は 2024 年度第 78 回文化祭にお越しいただきありがとうございます。

アマチュア無線研究部ではアマチュア無線のみに留まらず、ロボットプログラミング、電子工作、3D プリンターによる造形、3DCG など多岐にわたる活動を幅広く行っております。文化祭の展示ではこの一年間の活動の成果を見られるものとなっているので、楽しんでいただけすると幸いです。また、作品展示だけでなく電子工作教室やロボットプログラミング講習、無線通信体験なども行っているので、ぜひご参加ください！また、作品展示で分からぬことや気になったことなどあればぜひお聞きください。これからも灘校文化祭をお楽しみください！

(※電子工作教室とロボットプログラミング講習は事前の申し込みが必要です)

部員自己紹介

部員自己紹介

※()内の学年については文化祭時点での学年を表記しています。

82回生(中一)・新高 79回生(高一)

つい先月入学したばかりの一年生たちです！ 残念ながら自己紹介を載せることはできませんが、初めての文化祭ということで楽しみながら参加してもらえると嬉しいです。

77回生(高三)

・きゅうた(副部長)

77回生のきゅうたです。知り合いが見るのを考えると、ここでふざけるハードルってくっそ高いですよね。中1の自分とか、、見たくないです。今年は3年ぶりに作品を出したので、ぜひ見ていてください。
部誌を読んだらより楽しめるかも。

・yone

アマチュア無線にいるけど無線はできなくてロボットの人間です

78回生(高二)

・T.N.

主にロボットの大会に出てます
三年間出ているのですがまだ優勝できてないので次は優勝します

・cele

78回生の cele です。原神にハマってしまったため部誌が進みませんでした。

79回生(高一)

・T.R.(部長)

79回生の T.R.、部長です。2アマ試験のために勉強中です。無線サイコー！

・K.T.(会計)

俺は79回生会計、K.T。steamのセールを眺めるのに夢中になっていた俺は、背後から近づいてくる部誌の締め切りに気づかなかった。俺は balatro にどっぷりとハマってしまい、目が覚めたら…部誌が締め切られていた！！

・ちーしゃ

部誌執筆の運命を歩む物理属性のキャラクター

・そすいー

79回生のそすいーです。競技プログラミングしてます。

・キノコ

キノコです。今日も元気に生きてます。

部員自己紹介

・ T. K.

今から作品作ります(3/18)

Mac 最高！Mac 最高！

・ PiTaPa

79 の PiTaPa やで

三アマもっとるで

Blender やっとるで

左手デバイス開発中やで

80回生(中三)

・ よつばのくろおばあ

80回生の「よつばのくろおばあ」、ときどき

「みつばのくろおばあ」です。たまに「ふたばのくろおばあ」になります。

・ O. M.

3/24 時点でまだなにもできていません。

・ 眠い人

80回生の眠い人です。アマ研には去年から入りました。

最近体重があまり増えません。背は少し伸びました。

・ architecture

80回生の architecture です。

最近プログラミング、作曲、動画制作などしています。

いわゆる音ゲーが好きです。

・ BarrierReef

80回生の BarrierReef です。幽霊部員(部費は納めています)です。

普段はソフトテニス部に行ったり灘中入試算数の予想問題を作って

X(旧 Twitter)で公開したりしています。

アカウントの ID は@NADAtyuyosoubot です(完全なる宣伝)。

81回生(中二)

・ K. K.

81回生の K. K. です。実は4アマしかもってないんです、、、。

・ モ力

81回生のモ力です。最近炒飯が美味しいです

・ しん

いちおう3アマもってます

・ N

カービィ

ロケット

ロケット

yone

1 はじめに

最近(現在 2024/03/27 ←提出期限 over)ロケットに関する出来事がたくさんありました。今回ロケットや人工衛星などには無線がたくさん使われているのでアマチュア無線部の部誌のテーマとして選びましたが、僕は機械系の人間なので途中から無線の内容から脱線すると思いますので無線より機械の方が好きだという人も見て行ってください。(この部誌の中では主に日本のロケットについて取り上げます。)

2 ロケットの無線

ロケットには主にテレメータ送信機、レーダトランスポンダ、破壊コマンド受信機が乗っています。少ないと思った方もいるのではないかと思うが、現在ロケットは地上から制御されて飛んでいるわけではないので少ないのです。ではそれについて説明していきます。

テレメータ送信機について説明するうえでまずはテレメータについての話をしたいと思います。テレメータとは遠地の測定データを無線にのせて、監視センターに一定の時間間隔で自動送信することにより、監視センターで現地の状況をオンライン、リアルタイムで把握するために用いられる遠隔自動データ収集装置のことです。つまりテレメータ送信機とはロケットの現在の状況をリアルタイムで地上局において確認することをできるようにするというものです。ロケットの打ち上げの中継などを見たことがある人はその画面上に高度や速度、位置などの情報が映っているのを見たことがあると思いますがそれはこの仕組みによって送られてくるデータなのです。ちなみにテレメータは UHF 帯(860~960MHz の超短波)と S バンド(2-4GHz でアマチュア無線などでもよくつかわれている)を使用しており各段に乗っています。

次にレーダトランスポンダです。これはロケットを追跡するために使われています。先ほどのテレメータ送信機で十分と思われるかもしれません、ロケットや飛行機においてはあるものに不具合が起きても大丈夫なように様々なフェイルセーフシステムがあるのでそのためのものです。レーダというものは特別なものではなく、空港で飛行機の管制に使われている航空管制レーダや雨雲の動きをとらえるための気象レーダなど、多種多様なレーダが世の中に存在しています。これらのレーダは、地上の基地局から送り出した電波が目標物に当たり、その反射波を受けることで目標物の位置情報を得ていて 1 次レーダと呼ばれています。しかし、ロケットの追跡の場合目標が決まっているので他のものからの反射波は不要なわけです。そこで地上の呼び掛けに応えてくれる装置を、あらかじめロケットに搭載しておきます。次に、地上からコードを付した信号をロケットに送ってあげます。そうすれば、ロケットだけから返事が返ってくるという仕掛けになっています。この方法は 2 次レーダと呼ばれていて 1 次レーダと比べて遠くまで安定して追跡できるというメリットがあります。2007 年の JAXA の記事によると角度にして約 0.003° 、距離にして約 2m の精度でロケットの位置を瞬時にとらえることができるようです。

最後に破壊コマンド受信機です。ずいぶんと物騒な名前だと思われたかもしれませんがこれがロケットにおいて最も重要なもののひとつです。ここで問題ですがロケットにおいて最も信頼性が求められ、壊れてはいけないものとは何でしょうか？エンジンでしょうか、燃料でしょうか、それとも衛星でしょうかすべて違います。例えばこのような状況を考えてみましょう、エンジンが壊れて止まってしまったときや燃料が漏れてしまって燃料が足りないとなってしまったときロケットをどうするのでしょうか。H-IIロケットは300t(燃料込み)近くあるので制御できなくなってしまった場合は凶器でしかありません。ロケットを打ち上げる場合、切り離したものが落ちる予想となっている海域は閉鎖していますがそれ以外の海域には船が普通にいるのでそのような場所に落とすわけにはいきません。ここまで説明でわかった人もいると思いますが想定外の事態が起きて予定しているルートから外れることになった場合にはロケットを破壊させるのです。そのためロケットにおいて最も信頼性が求められ、壊れてはいけないものは破壊させるためのシステムなのです。これについてはあとで詳しく説明しますが破壊を判断した時に地上から送られた信号を受信するのが破壊コマンド受信機なのです。

3 人工衛星の無線

人工衛星に乗っている無線はそれぞれの人工衛星の役割によって種類がとても多いです。いくつか例を挙げると気象衛星として有名はひまわりでは

4. 通信回線の性能

4.1 可視赤外放射計による観測ミッション送信性能

4.1.1. 伝送フォーマット及び伝送レート

宇宙データシステム諮問委員会(CCSDS)の勧告に基づく伝送フォーマットを採用し、伝送レートは、 6.7 M s p s 以下である。

4.1.2. 使用周波数帯

送信K a 帯 $1.8 \sim 1.8, 4 \text{ GHz}$

4.1.3. 信号諸元

信号名	観測ミッションデータ
占有周波数帯幅	1.10 MHz 以下
送信E I R P	$6.2, 5 \text{ d BW}$ 以上
偏波面	円偏波又は直線偏波
変調方式	四位相偏移変調

※搬送周波数は、国際周波数調整後に決定される。

4.1.4. K a 帯送信アンテナ

- K a 帯送信アンテナはスポットビームアンテナとし、アンテナ照射領域は関東ビーム及び北海道ビームの2ヶ所に同時に送信する。
- 1つの送信ビームは、その中心点から半径 100 Km 内においては、規定 E I R P からの減衰量が 1.5 dB である。

4.2. 気象データ中継性能

4.2.1. 使用周波数帯

(1) 受信	UHF 帯 $4.02, 0 \sim 4.02, 4 \text{ MHz}$
(2) 送信	K a 帯 $1.8, 1 \sim 1.8, 4 \text{ GHz}$

4.2.2. 信号諸元

(1) 中継信号のアップリンクに関する諸元

信号名	通報局資料(報告)
受信周波数(注1)	$4.02, 0 \text{ MHz} \sim 4.02, 4 \text{ MHz}$
信号帯域幅	1.00 bps 信号: 2 kHz 以下 (1 波当たり) 3.00 bps 信号: 4 kHz 以下 (1 波当たり)
ダイナミックレンジ	-13.4 d BW/nf 以下 ～ -12.4 d BW/nf 以上
偏波面	右旋円偏波 (送信側から見て)
変調方式	パルス符号変調 / 位相偏移変調 ($\pm 60^\circ$)
受信 G/T	$-17, 0 \text{ dB/K}$ 以上

(注1) : 通報局資料(報告)は、 3 kHz 間隔で 133 波の信号を伝送する。

(2) 中継信号のダウリングに関する諸元

信号名	通報局資料(報告)
搬送周波数	$1.8, 1 \sim 1.8, 4 \text{ GHz}$
中継帯域幅	4.00 kHz
送信E I R P	$4.5 \sim 5.5 \text{ d BW}$
偏波面	円偏波又は直線偏波

※搬送周波数は、国際周波数調整後に決定される。

4.2.3. UHF 帯受信アンテナ

UHF 帯受信アンテナは、グローバルアンテナである。

4.2.4. K a 帯送信アンテナ

気象データ中継用送信アンテナは、観測ミッションデータに使用するアンテナと共に。

ロケット

4.3.1. 使用周波数帯

- (1) 送信
Ku帯 12200MHz～12750MHz
(2) 受信
Ku帯 13750MHz～14500MHz

4.3.2. 信号源元

- (1) 送信信号源元

信号名	Ku帯テレメトリー
搬送周波数	12200MHz～12750MHz の範囲
占有周波数帯幅	1100kHz 以下
送信 EIRP	スポットビームアンテナ： 20dBW 以上 無指向性アンテナ： 12dBW 以上
偏波面	スポットビームアンテナ：円偏波又は直線偏波 無指向性アンテナ：右旋又は左旋円偏波

*搬送周波数は、1 衛星につき 1 渡又は 2 渡使用する。なお、搬送周波数は、国際周波数調整後に決定される。

(2) 受信信号源元

信号名	Ku帯コマンド
受信周波数	13750MHz～14500MHz の範囲
信号帯域幅（注2）	1100kHz 以下
ダミエミッション	スポットビームアンテナ： -10.0dB 以下～-7.5dBW/m ² 以上 無指向性アンテナ： -8.6dB 以下～-7.5dBW/m ² 以上
偏波面	スポットビームアンテナ：円偏波又は直線偏波 無指向性アンテナ：右旋又は左旋円偏波

*搬送周波数は、1 衛星につき 1 渡又は 2 渡使用する。なお、搬送周波数は、国際周波数調整後に決定される。

4.3.3. Ku帯アンテナ

- Ku帯アンテナは、無指向性アンテナとスポットビームアンテナの 2 種類のアンテナを持つ。衛星異常時には自動的に無指向性アンテナに切り替わるものである。
- スポットビームアンテナは、九州から北海道（島嶼を除く）において、規定 EIRP 及び受信利得の低下は 1dB 以下。

気象庁 より

このようになっています。

また日本版 GPS とも呼ばれるみちびきのアンテナは特殊で、地表に届く電波の強さを均一にするヘリカルアレイアンテナというものが使われています。理由としては、アメリカの GPS やロシアの GLONASS や EU の GALILEO など、複数の衛星測位システムが同じ周波数(1.57542 GHz)を共用しているためそれぞれ衛星の信号の強さをある一定の範囲内ではほぼ均一にしなければいけないので、みちびきは他の測位システムと違って地表面との距離が変動する橿円軌道を通っているのでこの距離の変動も考えないといけなく、またほかシステムと比べて高度が高いためより強い信号を送信しなければいけません。地表面での信号の強さを均一に、そしてほかのシステムの信号よりも強く送るために 19 本のヘリカルアンテナで構成されるアレイアンテナを取り付けているのです。

このようにアンテナの種類も、周波数も人工衛星の役割ごとに違っています。

4 FTSについて

FTS(Flight Termination System)とは先ほど話にあがったロケットを破壊させるための仕組みのことです。主に指令破壊といわれ、地上で判断して行います。日本では種子島宇宙センター内の総合指令棟(RCC)においてこの判断が下されてコマンドが送信されます。指令破壊コマンドがロケットに届くと機体に搭載された指令破壊装置が起動して、火薬で燃料タンクを割ります。これは日本のロケットに共通する仕組みです。指令破壊の目的はロケットを安全に落とすために飛行を強制中断させる点にあります。そのため火薬で搭載された燃料に火をつけ爆破するわけではないのです。

ロケット

まあだからといって爆発しないのかといえばそんなことはなく爆発したように見えるのですが、爆破して粉々にするという目的ではないと認識してください。以下 FTS が作動したロケットたち



Starship (SpaceX/YouTube)

Alpha (Firefly Youtube)



カイロスロケット(読売新聞)

5 H3 ロケットについて

日本では H-IIA ロケットが主に使われておりこれは 2005 年の 7 号機から 40 機連続で打ち上げに成功しており、48 回中 47 回の打ち上げに成功しており打ち上げ成功率は 97.9%。H-IIA の強化型バリエーションである H-IIB ロケットも含めると 57 回中 56 回の打ち上げに成功しており打ち上げ成功率は 98.2% という高い成功確率のロケットです。しかし H-IIA ロケットは 50 号で終了することが決まっています。そしてその後継機が H3 ロケットというわけです。H3 ロケットの目標は JAXA の HP によると「今後 20 年間を見据え、毎年 6 機程度を安定して打ち上げることで産業基盤を維持するという運用の世界を目指しています。そのためには、政府の衛星だけでなく打ち上げサービス市場から民間の商業衛星の受注が不可欠です。

ロケット

世界中で新しいロケットが開発される中、商業衛星に利用してもらうためには、日本国内だけでなく世界中の利用者から使いやすいロケットとして注目されるような新しいロケットを作る必要があります。」とのことです。またそのために柔軟性、高信頼、低価格の3つの要素を実現するようです。それぞれのJAXAの説明は以下のとおりです。

柔軟性 (High flexibility)

複数の機体形態を準備し、利用用途にあった価格・能力のロケットを提供します。また、受注から打ち上げまでの期間短縮によるサービスの迅速化や、年間の打ち上げ可能機数を増やすことで、「迅速に打ち上げたい」という利用者の声に応えます。そのため、ロケット組み立て工程や、衛星のロケット搭載などの射場整備期間をH-IIAロケットから半分以下に短縮します。

高信頼性 (High reliability)

H-IIAロケットの高い打ち上げ成功率とオンタイム打ち上げ率（予定した日時に打ち上げられる率）を継承し、確実に打ち上がるロケットにします。

低価格 (High cost performance)

宇宙専用の部品ではなく自動車など国内の他産業の優れた民生品を活用するとともに、生産の仕方についても受注生産から一般工業製品のようなライン生産に近づけることで、打ち上げ価格を低減させます。固体ロケットブースタを装着しない軽量形態（主に低軌道の打ち上げに用いる想定）で約半額を目指しています。H3ロケットの開発費は約2061億円(H-IIAはH-IIからの改良開発費だが約1532億円)打ち上げ費用は約50億円(H-IIAは85~120億円)軌道投入能力は太陽同期軌道(500km)には4t以上、静止移行軌道($\Delta V=1500\text{m/s}$)には6.5t以上(H-IIAは以下の図JAXAのHPにあるので見えない場合はそちらを見てください)

搭載	全長	標準型		高度化仕様		
		H2A202	H2A204	H2A202	H2A204	
		4S (Φ4mフェアリングFairing)		53m		
重量		269t (人工衛星の質量は含まず)		慣性挾持方式		
搭載	標準静止トランスポンダ軌道 離地点高度 36,226km 近地点高度250km 軌道傾斜角28.5度 静止化燃耗量 ΔV $+1,830\text{m/s}$	4,000	5,950	—	—	
打ち上げ能力	ロングコスモ静止トランスポンダ軌道 離地点高度 36,226km 近地点高度 2,700km 軌道傾斜角20度 静止化燃耗量 ΔV $+1,500\text{m/s}$	—	—	2,970	4,820	
搭載	太陽同期軌道 高度800km 軌道傾斜角98.6度	3,300	—	—	—	
搭載	高度300km 軌道傾斜角30.4度	10,000	—	—	—	

*ペイロードアダプタ質量を10kgと仮定

このようにH-IIAと比べて高性能で

打ち上げ費用が安くなっています。

ロケット

太陽同期軌道(SSO)とは太陽と太陽同期軌道とは位置関係が年間を通じて毎日同時刻に一定となっていて衛星から地球を見ると太陽光の入射角が常に同じになり同一条件下での地球表面の観測が可能となる軌道のことと、静止移行軌道とは人工衛星を静止軌道にのせる前に一時的に投入される軌道のことです。

ここまでH3ロケットの目的や性能について書いてきました。ここからは実際に打ち上げられたH3ロケット試験機1号機、2号機について書いていこうと思います。

H3ロケット試験機1号機はだいち3号を運ぶ予定でした。だいち2号は2014年に打ち上げられて設計寿命は5年で目標7年の衛星でしたが後継のだいち3号がいろいろあって打ち上げられておらず、ずっと運用が続いていました。

H3ロケットの初飛行は元々2020年に予定されていましたが第一段主エンジンをH-IIAで使っていたLE-7AからLE-9に切り替える開発が難航したため2度延期されて2023年3月7日になりました。(実はまだエンジンの開発は終わっておらずまだまだ開発途中です。)

そしてついに打ち上がったH3ロケット試験機1号機ですが第二段ロケットが燃焼開始せず、回復する見込みがないとして指令破壊となりだいち3号も失われてしまいました。

すぐさまテレメータデータを解析するなどして原因の特定が始まりました。指令破壊を行った時点で高度400kmを超えていたので残骸を回収しておらず理由をひとつに断定することはできなかったのですが、3つのシナリオを考えてそのすべてを対策することで対応しました。

1つめのシナリオとしてはエキサイタ(電気から火花を作る装置)が何らかの原因でほかの場所に接触し、そちらに電流が行くことによって壊れてしまったというもので、絶縁体で保護することで対策しました。これはH-IIAと共に問題でH3ロケット試験機1号機の打ち上げの後H-IIAにも同じ対策を施しています。

2つめのシナリオとしてはトランジスタの耐圧を超える電圧が試験の時点でかかっていて打ち上げのタイミングで壊れたというもので、あとで試験の時に耐圧以上の電圧がかかっていることがわかりました。これについては設計を見直して対策しました。これもH-IIAと共に問題でH3ロケット試験機1号機の打ち上げの後H-IIAにも同じ対策を施しています。

3つめのシナリオとしてはPSC2(2段推進系コントローラ)を冗長性を持たせるためにA、Bの2系統用意していたがAでの異常により過電圧が生じてAが遮断され、定電圧ダイオードがショートし、電源のリターンラインを経由してBに伝搬してBも遮断されてしまってエキサイタに電源供給ができなくなってしまったというものです。この対策としては定電圧ダイオードは、PSC2の過電圧検知遮断機能に加え、下流機器を保護する目的で実装していたものであるが取り除いても問題ないことが確認できたので取り除くことをしました。これはH3ロケット固有の問題でした。

これらの対策をしたうえで2024/02/17にH3ロケット試験機2号機を打ち上げました。このとき試験機1号機に近づけるために重心の位置や重さを再現したおもりとおまけで超小型衛星2つを搭載しました。

ロケット

結果として成功しこの超小型衛星 2 つが H3 ロケットで初めて軌道に投入された衛星となりました。(この時 2 つある第一段主エンジンのうち 1 つを変えたりもしています。)

この章のもっと詳しい話はこれまた JAXA の HP に載っているのでぜひそちらをみてください。

6 カイロスロケットについて

カイロスロケットは IHI エアロスペースが元となったスペースワン社が開発した日本初の民間で軌道投入することができるロケットです。このロケットは固体燃料を 3 段採用し液体燃料は最後の 1 段にのみ採用していて、また FTS をロケットが自分で判断して行います。そのため効率的ですぐに打ち上げができるようになっていてロケット市場での競争力を得ようとしています。

こちらは当初 2021 年度中の打ち上げを目標としていましたが新型コロナウイルス感染症の蔓延やロシアによるウクライナ侵攻の影響で部品調達が遅れ、4 度にわたって延期されました。そして 2024 年 3 月 11 日にスペースポート紀伊(和歌山県串本にある)の初めての打ち上げとして打ち上げられました。今回、すぐに打ち上げられるという特性を生かして内閣衛星情報センターの短期打上型小型衛星を搭載していました。これは情報収集衛星で、今使っているものに何らかの異常が発生し、使えなくなった時のピンチヒッターとしての役割を担うものとなっておりすぐに打ち上げられることがとても重要なものです。

しかし残念ながらカイロスロケットの初飛行は打ち上げのおよそ 5 秒後に FTS が自動的に作動し爆発しました。FTS が作動したロケットのところにカイロスの名前があったのでどうなったかわかっている人も多いと思います。

原因は書いている現在まだわかりませんが原因が特定でき、それに対処して 2 回目の打ち上げが行われるのを楽しみに待とうと思います。高 3 になるので見れないかもしれませんけど…

7 さいごに

今回の部誌では主に日本のロケットについてしか取り上げておらず、しかも例えばイプシロンロケットや M-V ロケット、H-I・H-II ロケットなどの H-IIA、H3、カイロス以外のロケットにの話もできていないし、衛星の話や日本初月面に軟着陸を成功した SLIM などの話もできていません。海外のロケットについては最近 Starship が 3 回目の試験打ち上げを行い大気圏突入時に分解はしましたがそれでも素晴らしい成果を得るなど日本よりも盛んにロケット開発やロケットの打ち上げが行われています。この部誌をここまで読んでいただき、ロケットや宇宙に興味が沸いたらその内容について詳しく調べてみたり、図書館などで本を探して読んでみたりしてください。

ロケットに少しでも興味が沸いたら宇推くりあさんという Youtuber(Vtuber) をぜひ見てみてください打ち上げの中継を Live でよくされていてその中でロケットについての解説をわかりやすくしているので初めて見ても色々なことを知れてたのしいと思います。

ロケット

H3 ロケット試験機 1 号機、2 号機、カイロスロケット、Starship…と今回紹介したような最近のものはアーカイブがあるので見てみてはいかがでしょうか。ちなみに JAXA の公式配信に出ていたり SLIM、H3 ロケットの応援サポーターだったり内閣府宇宙開発利用大賞 PR キャラクターだったりとすごい方です。

ほとんど無線の話はなかったですが僕はロボットをやっている側の人間なのでご容赦ください。これを読んでロケットや宇宙に興味を持ってくださった人がいれば徹夜で 6 時まで書いたのも報われます。長くなりましたがここまで読んでいただいたかたありがとうございました。

学校から配布された端末でゲームをする話

学校から配布された端末でゲームをする話

ちーしゃ

この記事では実際に灘中学校から配布された端末、Surface Go でゲームを遊ぶにはどうすればいいかについて書きます。突然ですが皆さん、教育用の端末でゲームをしたいと思ったことはありますか？~~ありますよね？そう、あるんですよ。~~

具体的にやること

Surface Go にインストールされた Windows 10 を丸ごと吹き飛ばし、代わりに Bliss OS をインストールします。Bliss OS とはなんぞやという話ですが、簡単に言うと Android です。iPhone 以外のスマホのあれ。正確にいえば Bliss OS は Android をスマホではなく PC にインストールして使うためにいろいろ改造したものです。（改造とは言っても合法です、大丈夫）つまり Surface Go に Bliss OS をインストールするということは、Surface Go を Android タブレットにすることです。

なぜ Bliss OS なのか

なんでわざわざ Android なのか、それは本来 Surface Go にとって Windows 10 は荷が重い代物だからです。そもそも一般的に PC のスペックはスマホよりも高いです。やはり精密機械を小さくするというのは大変で、そこに労力を注ぐ以上、処理能力には制約が出てきます。逆に言えば PC にはスマホよりも余裕があるので、多少重たいプログラムでも動かすことができます、一般的な PC なら（ここ重要）。ここで我らが Surface Go に話を戻すと、この子は数年前の製品、さらに本体の軽量化に全力を注いでおり、値段もそこまで高級機種という訳ではないので、現代においては平気でスマホに引けを取ります。もはや PC と呼ぶことはできません、こいつはただ Windows 10 を積んだだけの板です。しかしそれも Windows 10 での話、それよりも負荷の軽い Android ならばまだ…という一縷の望みにかけて、というのが今回 Bliss OS で記事を書いている理由です。そもそも PC 用のゲームを遊ぶのなら記事もへったくれもありません、灘中学校から配布された端末に特に制限などはかかっていません。もっとも口が裂けても快適なゲーム体験とは言えないでしょうが…

必要なもの

実際の作業で必要なものです。最低限必要なものは Surface Go 本体と充電用ケーブル、USB-C 接続の USB メモリです。USB-C でないといけません、百均なんかでは多分売っていないと思います。これも全部 Surface Go の拡張性の低さの弊害です。容量はそんなに多くなくても大丈夫です、16GB 以上なら大丈夫です。ただし作業の途中で中身はすべて吹き飛びます。中に大事なファイルがある USB メモリではこの作業はやらないでください。もしそれが原因でなにかまずい事態になっても私は責任を負いません。私は責任を負いません。大事なことなので二回言いました。必須とまでは言いませんが USB-C が使えてまともに windows(10 か 11) が動く PC があれば嬉しいです。

学校から配布された端末でゲームをする話

Surface go 自体でも作業は出来るといえばできますが、操作性が悪いですし、何よりどこか間違えて surface go が何の変哲もない金属の板と化した場合に完全に詰んでしまいます。どうしてもほかに使える PC がないという方はあきらめるか、学校から課された課題がこの先すべて提出できなくなるハンデを背負う覚悟を決めるかの二択です。個人的には前者がおすすめです。成績、大事。

作業

最初にいろいろとダウンロードするものがあります。まずは Bliss OS 自体のデータ。「Bliss OS」で gg れば公式サイトが出てきます。下にスクロールすると「Bliss OS Downloads」が見えてくるので、使いたいものを選びましょう。今回は Bliss OS 15 を選択した後、一番上の「Bliss OS 15.9.x (x86_64-v2) with GApps」を選びます。下のほうに surface 用バージョンとかいうものが置いてありますが、自分の端末では正常に動作しませんでした。SourceForge と書かれたボタンを押すと寄付をねだられますが、少し待つとそのまま次のページに進みます。上から二番目の「Bliss-v15.9-x86_64-OFFICIAL-gapps-20240114.iso」をクリックしてダウンロードしましょう。かかる時間は結構長いです、気長に待ちましょう。

次に必要なソフトがあるのでこれもダウンロードします。「rufus」で gg って、rufus の公式サイトから「rufus-4.4.exe」をダウンロードしましょう。

そうしたらインストールメディアを作ります。インストールメディアとはなんぞやですが、USB メモリに特別なファイルを書き込んだものです。ブータブル USB(ブータブルは英語で起動可能という意味)とも呼ばれます。つまり起動可能な USB です。…起動可能ってなんぞや。PC が起動するためにはまず OS の情報を読み込まなければいけません。起動可能というのを平べったく言うと、OS の情報が丸ごと書かれているということです。

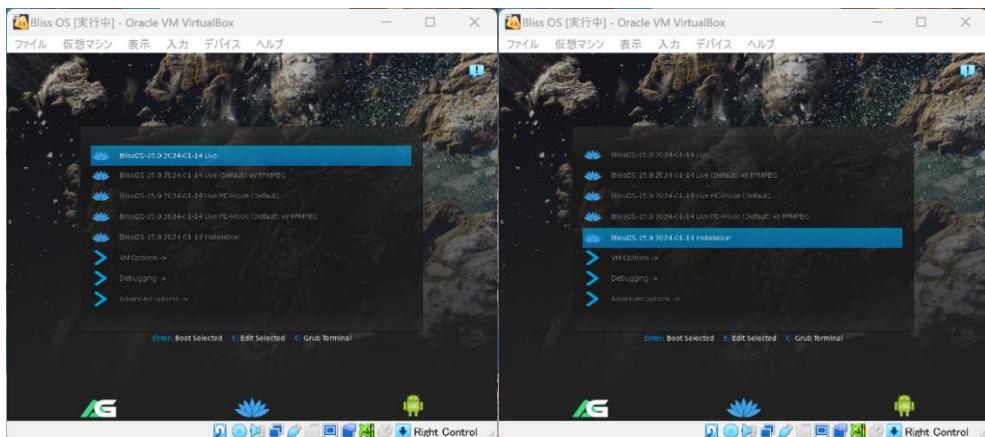
まずは rufus を起動します。先ほどダウンロードした rufus のファイルをダブルクリックしましょう。無事に起動出来たら USB メモリを挿してファイルを書き込みます。デバイスの欄で USB メモリを選択したら、その右下の「選択」からさっきダウンロードした Bliss OS のデータを選択し、「スタート」を押しましょう。準備完了したら rufus は閉じて、USB メモリを抜きましょう。

次は surface go の準備です。ここから先は AC アダプタで給電しながらだと安心です。まずは surface go の電源を入れて Windows10 を起動しましょう。設定、回復、PC の起動をカスタマイズするの順番に進み、今すぐ再起動を押します。しばらくすると真っ青なオプション選択画面が出てくるので、トラブルシューティング、詳細オプション、UEFI ファームウェアの設定の順に進みます。再起動を押すと UEFI の設定画面が出てきます。上から二番目の Security を選び、Secure Boot の欄を Disabled にしましょう。これで Surface 本体の準備は完了です。Exit から Restart now を押しましょう。Microsoft ちゃんが赤色の警告で脅してきますが、気にしたら負けです、慣れましょう。

学校から配布された端末でゲームをする話

いよいよインストーラーを起動します。とはいってもまだインストール 자체はせず、正常に動くかのテストを行います。なお、ここからの画像は別の環境を使った再現画像です。実際のものとはちょこちょこ違う部分があります、気を付けてくださいね。Windows10が起動したら、USBメモリを挿し、一度通常の再起動をした後にもう一度回復、PCの起動をカスタマイズするから今すぐ再起動を押します。今度はデバイスの使用を選び、Linpus liteを選択しましょう。田んぼマークが一瞬見えた後、インストーラーが起動します。いろいろ選択する画面が出てきますが、そのまま放っておくとライブ起動というモードで起動します。OSをインストールしなくとも試しにちょっと使えるというもので、一度は試しておいたほうが安心です。ライブ起動をすると、Bliss OSのロゴが見え、しばらくすると通常のAndroidと同じセットアップ画面が表示されます。タッチ操作やキーボード入力が正常に行えるか試しましょう。ただし、キーボードの配列は正常には読み込まれませんでした。

Surfaceのキーボードは特殊なので、例えば@を入力しようとすると、本来の@の場所ではなくShift+2を押さなくてはいけません。ドライバを頑張って自力で突っ込めるなら話は別ですが、そこまで物理キーボードにこだわらない方はそのまま使うのが楽でいいと思います、私はそうしました。正直Androidである以上スクリーンキーボードで事足ります。



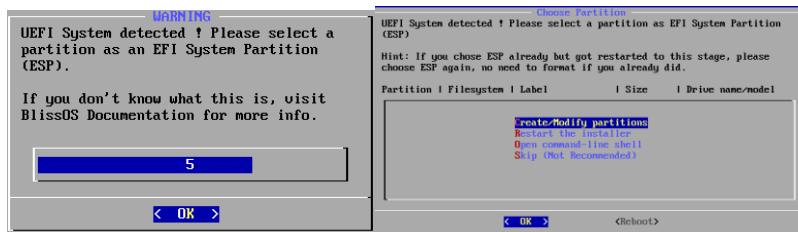
それでは実際に書き込みましょう。もう一度インストーラーを起動したら、↓を数回押して、下のほうの「Installation」を選択してエンターを押します。ここからは私の環境で起こったことなので、もしかするとここに書くこととは違うことが起こるかもしれません、そんなときは頑張って英語を読んで自力でインストールするか諦めてください。下手にいじくりまわすよりは諦めるほうがまだましです。

「EFIが認識されたよ！」的なことを言われるので、エンターを押し、EFIシステムパーティションの場所を選択します。(EFIシステムパーティションとは何かについて解説し始めると文章量が壊れてしまうので、ここではそういうものだと思ってください。詳しく知りたい人は自力でggってください。)USBメモリとSurface go自体のストレージの二つが表示されるので、本体のほうを選択して「Create/Modify partitions」を押します。

学校から配布された端末でゲームをする話

容量が 64GB の方が Surface go 本体です。USB メモリの容量も 64GB だった場合は…気合で頑張りましょう。エンターをもう一度押すと cfdisk が起動します。cfdisk はストレージのあれこれを削除したり作成したりするツールです。まずは「free space」と書かれた場所以外、全て削除しましょう。上下の矢印キーで削除したい部分を選択したら、左右のキーで「Delete」を選択してエンターを押します。これを繰り返し、一つの大きなフリースペースを作ります。そうしたら EFI システムパーティションを作ります。「free space」を選択していることを確認して、下の「New」を選択します。サイズを聞かれるので「512M」と入力してエンターしましょう。次にタイプを変更します。さっき作った 512MB の領域を選択していることを確認したら、下の「Type」を押します。ずらっとタイプ一覧が出てくるので、一番上の「EFI System Partition」を選択しましょう。そしたら最後に Bliss OS 自体を書き込む領域を作ります。残っているおおきなフリースペースを選択して New を押します。サイズは最大のままなので、今回は何も入力せずそのまま作成します。タイプの変更もしないで大丈夫です。これで cfdisk での作業はおしまいです。

最後に作業の内容を実際に反映します。下の「Write」で作業を書き込みましょう。エンターすると確認が出てくるので、yes と入力してもう一度エンターします。これで準備は完了、「Quit」で cfdisk から出ましょう。インストーラーに戻るともう一度 EFI システムパーティションの場所を聞かれるので、先ほど作成した 512MB の領域を選択します。次に Bliss OS 自体を書き込む場所を選択します。フォーマットの種類を聞かれるので ext4 を選択します。フォーマットとはファイルを書き込む方式のことです。ストレージにファイルを書き込む方法には実はいろいろあって、android に使われる ext4 や Windows にしか使えない NTFS、様々な OS で使える FAT32 などがあります。次に OTA アップデート云々を聞かれます。よくわからないので自分はとりあえず YES って答えてます。よくわからないのは基本的に YES でいい気がします。Surface go はぶつ壊れたらその時はその時です。次に GRUB をインストールします。GRUB はブートローダーと呼ばれるソフトウェアの一つです。詳しい説明は省略しますが(というより自分も完全には理解していない)、OS を起動させるのに必要なソフトウェアです。なんか GRUB っぽい選択肢が出てきたらとりあえず yes を押せば何とかなります。最後にブートローダーをインストールするパーティションをフォーマットしたら、いよいよファイルの書き込みが始まります。あとは待ちましょう。最後まで完了したら、reboot を押して再起動し、Bliss OS が起動したら USB メモリを抜きましょう。ちゃんと起動しなかったり、USB メモリを抜いたとたんに画面が暗くなったりした場合は…ご愁傷さまです(一敗)



学校から配布された端末でゲームをする話



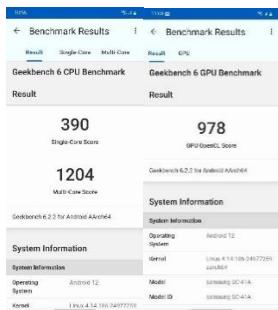
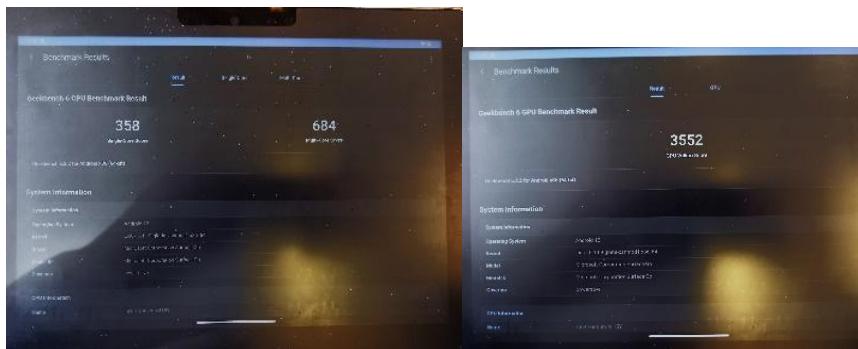
実際の動作

さて、実際に性能を試してみましょう。見せてもらおうか、Android 搭載 Surface go の性能とやらを…まずはいろいろベンチマークを回します。ベンチマークとは、決まった処理をさせてその結果を見ることで、その機械の性能を数値化できるソフトです。簡単に言うとテストみたいなものです。それでは最初はスマホ用ベンチマークの鉄板、AnTuTu Benchmark…と行きたいところですが、なんだかよくわからないですが正常にテストを完了できませんでした。かなしい。エラーメッセージも何も出してくれないので解決も難しそうです。

学校から配布された端末でゲームをする話

なので今回は諦めます。というわけで次、こちらもベンチマークの定番、GeekBench。実際に測定した結果がこちらです。数値だけ出されてもどんなものかわからないので、比較対象として私のスマホ(Galaxy A41)でのテスト結果も掲載します。ちなみにけっこう時間がかかります。

上が Surface go のスコア、下が Galaxy A41 のスコアです。Surface go の CPU のスコアはシングルコアが 358、マルチコアが 684、GPU のスコアは 3552 です。それでは比較対象も見てみましょう。Galaxy A41 の CPU のスコアはシングルコアが 390、マルチコアが 1204、GPU のスコアは 978 です。こうして見てみると CPU のスコアでは私のスマホが勝っていますが、GPU スコアでは Surface go にぶっちぎられています。なんででしょうね。正直実際にゲームをしていてそこまで性能が高いとは思えなかったので、なにかしらテストに異常が起ったのでしょうか。まあ AnTuTu は完走すらできなかつたわけですし、あまりスコアは信用せずに行きたいと思います。



学校から配布された端末でゲームをする話

それでは実際にゲームで遊んだ際に感じた体感の性能についても触れておきましょう。まずは動くゲームについて、起動すらできないほどに動かないゲームはそんなにありませんでした。比較的軽めのゲームなら問題なく遊べると思います。特に3Dデータを一切扱わないようなシンプルなゲーム(例えばにゃんこ大戦争など)は快適に動作しました。そこまで重いゲームは遊ばないという場合なら十分楽しめると思います。リズムゲームではさすがにラグが目立つシーンもありました。Phigrosのような凄まじく軽いリズムゲームならカクつく様子もありませんでしたが、プロジェクトセカイのような比較的リッチな演出を使ったゲームではカクつきが見られました。最後に激重ゲームも試しておきましょう。大人気ゲームにして同時にスマホ最重量級のゲーム、原神です。しかし、ここで問題発生。なんかGoogle Playが「対応していません」とか文句言ってくるのでインストールできません。しょうがないので同じ会社のこちらも激重ゲーム、崩壊・スターレイルで試してみると、インストールこそできるものの、起動すると読み込みの途中で落ちてしまいます。さすがに荷が重すぎたのでしょうか。ただし崩壊3rdは動いてくれました。どうやら3Dゲームでありながら、一回り負荷が小さかったようです。ただし少し触ると落ちてしまいそうなほど動作は不安定なので、まともに遊べているとは言えないかもしれません。崩壊学園は2Dゲームなので多分動きます、試してないけど。~~あっれー原神くん、どうやら君の兄妹さんたちはみんなSurface goでも動いてくれるみたいだけど、どうしちゃったのかなー??あと最後に補足することとしては、Surface goは排熱用の機構を一切持たないので、すぐに熱を持つし温度が上がると途端に性能が下がります。カバーを付けたり、足の上において使うと動くものも動かないでの気を付けましょう。~~

あとがき

機動戦士ガンダム SEED FREEDOM面白かったですね。私はガンダムSEEDシリーズ過去作をしっかり見てるわけではなくて、特に映画館に行った時点ではYouTubeに上がってる公式のまとめ動画(SEEDとSEED DESTINY合わせて一時間ちょっと)とスペシャルエディション(もうちょっと中身あるやつ、SEEDとSEED DESTINY合わせて十時間半)のSEEDの部分の2/3、時間にして三時間分しか見れていなかったので、内容が全部理解できたとは到底言えませんでしたが、それでもハチャメチャでとても楽しく見ることができました。やはりロボット物はいいですね、ロボットがぐりぐり動いてるところを見るだけでなんだか幸せです。家にプラモデルを積んでしまっているというのについてデスティニーガンダムが組みたくなってしまいます。そりやああんなかっこいいとこ見せつけられたらそりやあ…ねえ?あとシンかわいい。DESTINYは全然知らないのでもちろんシンがどんな子かも知らないのですが、劇場版のシンは本当にかわいかつたです。なんなら映画で一番。早くDESTINY見なきや…

学校から配布された端末でゲームをする話

あとがき

今回の記事、いかがだったでしょうか。実際に試してみたという人、失敗しても文句言わないでくださいね。今回の記事でOS周りに興味を持ってくれた方、一つだけ言いたいことがあるとすれば、とにかくいっぱい gg ってください。機械周りで何か気になることができたとき、本も悪いとは言いませんがやはり検索してしまうのが一番です。最後に、拙文だつたにも関わらずここまで読んでいただきありがとうございました。

執筆中 BGM

See-Saw 「あんなに一緒だったのに」

See-Saw 「去り際のロマンティクス」

STEAKA 「ドーピングダンス」

夏山よつき 「ネクラヒリア」

ベートーベン 「ピアノソナタ 月光 第三楽章」

ベートーベン 「ピアノソナタ テンペスト 第三楽章」

BlackY 「Alea jacta est!」

USAO 「Chariot(Extended mix)」

パガニーニ/リスト 「ラ・カンパネラ」

遺伝的アルゴリズム

遺伝的アルゴリズム

そすいー

遺伝的アルゴリズム(GA: Genetic Algorithms)は実際の遺伝子を元に考案されたアルゴリズムです。主に「目的はわかっているが、手段はわからない」というときに使用されますが、遺伝的アルゴリズム自体は非常に汎用性が高く、様々な問題に適応可能です。このアルゴリズムは生物の進化の仕組みを模したアルゴリズムで、随所に生物的な考え方が垣間見えます。このアルゴリズムの大まかな流れとしては、まずランダムに解を生成し、評価の良いものを取り出し、それらを元にランダムに解を生成し…ということを繰り返し最適解を導き出します。ここで我々がまず考えなければならないのは「解をどのように設定するか」と「評価の仕方」です。

解をどのように設定するかですが、主に配列として{パラメータ1、パラメータ2、パラメータ3、…}というような感じで設定しているものが多い気がします。また「評価の仕方」は評価関数と呼ばれ、ここをどのように決めるかで結果や効率が大きく異なります。

遺伝的アルゴリズムの考え方を理解するうえで OneMax 問題というものが分かりやすいと思ったので次節では OneMax 問題を見ていくたいと思います。

OneMax 問題

OneMax 問題とは0と1からなる数列をすべて1にする問題のことです。つまり

$$\{0, 1, 1, 0, 0, 0, 1, \dots\} \rightarrow \{1, 1, 1, 1, 1, 1, 1, \dots\}$$

とする問題のことです。便宜上、ここでは最初の数列は要素がすべて0であるとします。

まず、解をどういう風に設定するかですがこれはそのまま数列を解の表現方法として採択するのがいいと思われます。今回でいえば数列の各要素がそのまま遺伝子の各要素になります。また、評価関数をどのように設定するかですがこの場合は数列の和をそのまま得点とするのがよいでしょう。この評価関数を様々に変えることで問題に応じた最適解を見つけることが可能となります。

今回は両親から子を生成するときは50%の確率で父、母の各要素をコピーしています。しかし、優れた方の遺伝子を取りやすくするのが一般的なようです。どちらからどれだけの割合でとるかを決める方法としてランキング方式、ルーレット方式などがあるようです。

子に突然変異を起こさせるのも重要です。一般的には実際の遺伝子に倣ってか二点交叉という方法を用いることが多いようですが、今回は各要素に対し1%の確率で0と1を反転させています。状況や問題に応じて使い分けるとよいと思われます。

親から子を生成させる回数を世代という単位で数えますがこの世代は多ければ多いほど最適解に近づきやすくなります。以下はC++による実装例です。

```
#include <bits/stdc++.h>
using namespace std;
```

```
#define SZ 10
#define no_of_gene 10
#define generation 10000

// 確率を計算するためのプログラム
random_device rd;
mt19937 gen(rd());
int probability(int low = 1, int high = 10000) {
    uniform_int_distribution<> dist(low, high);
    return dist(gen);
}

// 遺伝子たち
vector<vector<int>> gene(no_of_gene, vector<int>(SZ));
// スコア計算
int score(const vector<int> & vec) {
    int ret = 0;
    for(auto && it : vec) ret += it;
    return ret;
}

// 混ぜる
vector<int> mix(const vector<int> & parent1, const vector<int> & parent2) {
    vector<int> child(SZ, 0);
    for(int i = 0; i < SZ; i++) {
        // 親たちを混ぜる
        if(probability() <= 5000) child[i] = parent1[i];
        else child[i] = parent2[i];
    }
    return child;
}

// 突然変異
void mutation(vector<int> & id) {
    for(int i = 0; i < SZ; i++) {
        if(probability() <= 100) id[i] = (id[i] + 1) % 2;
    }
}

signed main () {
```

遺伝的アルゴリズム

```
for(int i = 0; i < generation; i++) {
    int P1 = probability(0, no_of_gene - 1), P2 = probability(0, no_of_gene - 1);
    vector<int> child = mix(gene[P1], gene[P2]);
    mutation(child);
    if(score(gene[P1]) < score(gene[P2])) gene[P1] = child;
    else gene[P2] = child;
}
cout << "results" << endl;
for(int i = 0; i < SZ; i++) {
    for(auto && it : gene[i]) cout << it << " ";
    cout << endl;
}
return 0;
}

// ここまでが実装例です。
```

OneMax 問題は非常にわかりやすいですが、ここから少しコードを変えるだけでナップザック問題なども解けたりするようです。注意点として、突然変異の割合には気を付けなければならないようです。変異を起こしやすすぎると解が一つに定まりにくくなったり、起こしにくすぎると局所解にはまってしまったりするそうです。

最後まで読んでいただきありがとうございました。（誤りがあるかもしれません、ご了承ください。）

ラジオの仕組みと自作の方針

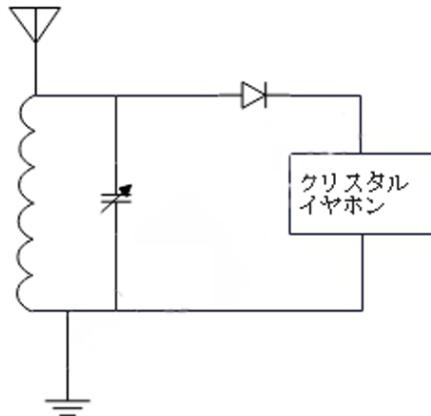
T. R.

79回生のT.R.です。突然ですが塹壕ラジオ（もとい鉱石ラジオ）というものはご存じでしょうか？これは第二次世界大戦中に兵士が娯楽目的で放送を聴くために作られました。材料はエナメル線と鉱石（いい塩梅に半導体の代わりになるもの、例 黄鉄鉱、方鉛鉱）、炙った剃刀の刃などの比較的手に入りやすいものなので、戦時中にも作ることができたそうです。では電源供給はどうしていたのでしょうか？当然コンセントや電池のようなものを用意することはできなかったでしょう。しかし、なんと、このラジオの最大の特徴は電源供給が必要なくラジオ局からの電波だけで動くことなんです。いいネ！塹壕ラジオはラジオの基本形と言えるでしょう。構造もかなり分かりやすいので現在でもよく自作されています。

しかし鉱石を使って自作となると鉱石の品質に左右されるのでそこそこ手間がかかってしまいます。

ここで文明の利器（そもそもラジオが文明の利器ではあります）ゲルマニウムダイオードを鉱石の代わりに使ったゲルマニウムラジオを紹介したいと思います。

ゲルマニウムラジオの回路図

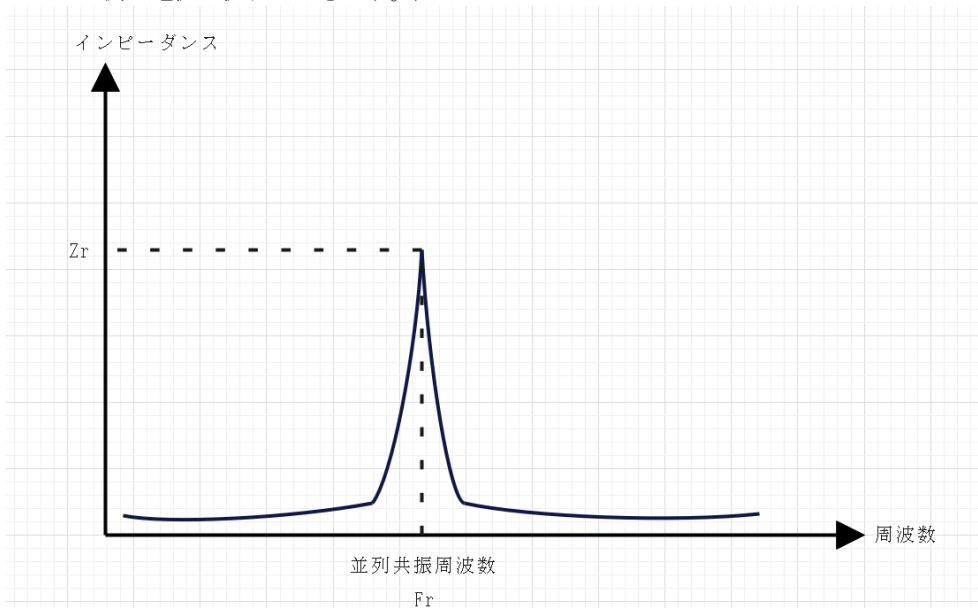


回路はアンテナ、コイル、ポリバリコン、ゲルマニウムダイオード、アースクリスタルイヤホンから構成されます。

〈仕組〉アンテナからの電波を受信→コイルとポリバリコンの同調回路で周波数を選択→ゲルマニウムダイオードで交流電圧の負の部分を除去（整流回路）→[ダイオードは基本決まった方向にしか電流が流れない]→クリスタルイヤホンで電流を平滑にする。→クリスタルイヤホンから音が出る。

ラジオの仕組みと自作の方針

同調回路は LC 並列共振回路というものが使われており周波数は次の式で表されます。
並列共振周波数 $f = 1/2\pi\sqrt{L/C}$ (L はコイルのインダクタンス、 C はコンデンサの静電容量)
この周波数のときに LC 並列回路のインピーダンス(平たく言えば抵抗)が最大になるので
回路に電流が流れます。(並列共振周波数以外の時はインピーダンスが 0 に近くなっているの
でアース側に電流が流れていきます。)



クリスタルイヤホンを使う場合それ自体が静電容量を持っています。よってコンデンサのようなふるまいをするので、平滑回路の役割を果たします。(平滑回路とは整流された電流を滑らかにして直流にする回路のことです。)

まとめると受信した電波に同調と整流と平滑化を施すことで音としてクリスタルイヤホンから出力できるということです。

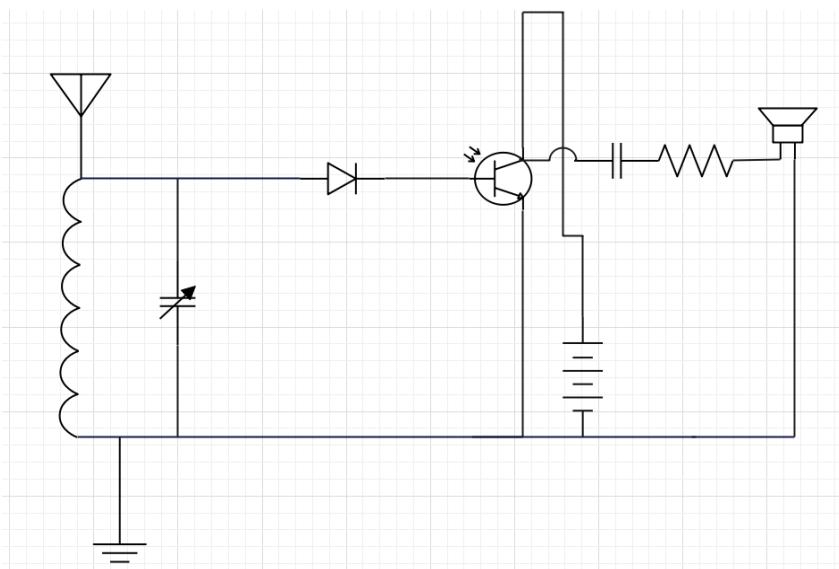
僕は「こいつ（ゲルマニウムラジオ）を自作して文化祭用に展示してやろう。」と思った訳ですがここでいくつかの問題に気づきました。

一つ 文化祭で展示するのにイヤホンを使うのは衛生的に良くない気がするということ
二つ 前述したクリスタルイヤホンは日本では希少で手に入れるのが割と難しいといいうこと。
(特にロッシェル塩を使っているものは湿気に弱いので保存するには注意が必要。その上現在日本では生産停止している。セラミックを使ったものはまあ探せば見つかると思う)

これらの問題をどう解決したものかと考えました。(考える人のポーズ) 考えた結果がこちら。
整流した電流をトランジスタで増幅させ普通のスピーカーでも音が鳴らせるようにしよう！

ラジオの仕組みと自作の方針

回路図



変更点

- ・クリスタルイヤホン→スピーカー
- ・NPNトランジスタ追加
- ・抵抗追加
- ・コンデンサ追加(平滑のため)
- ・増幅するための電池←!?

↑

電源を必要としないゲルマニウムラジオに電池を必要とするギャグ(多分新喜劇なら転んでる)

この回路図ですがまだ考えただけでシミュレーションや計算とかはしていないので間違っているかもしれません。

僕が懸念しているところはトランジスタがエミッタ接地だと入力電圧と出力電圧の位相がずれるので音声が乱れるのではないかということです。

位相がずれているので予想だと音声に遅延が少し発生するんじゃないかなと思っています。

ここから計算をして部品を買いに行って制作をしようと思います。

このような感じでラジオ自作の方針を立てることができました。

以上で文章を終わります。最後まで読んで頂きありがとうございました。

Live2D でキャラクターを動かす！

Live2D でキャラクターを動かす！

T. R.

79回生のT.R.です。僕は趣味で絵を描いているのですが、ある日オリジナルキャラクターを動かしたくなり調べてみたところ Live2D というものがあることを知りました。

今回 Live2D を使ってみたので、最後まで読んで頂けると嬉しいです。

①立ち絵作成の流れ

立ち絵は FireAlpaca というアプリを使って描きました。

ラフ→線画→色塗り→パーツ分けという順に行います。

(1)ラフ

まずラフを正面から描きます(図①)。板タブを使いました。

(2)線画

(1)で描いたラフをもとに線画を描いていきます(図②)。この時点である程度後からする
パーツ分けのことを考えながらレイヤーを整理しておきます。

(3)色塗り

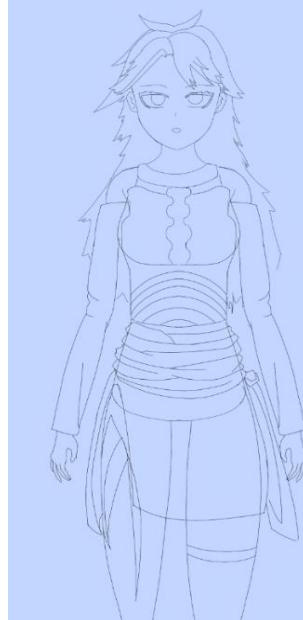
紙の部誌はモノクロなので色が分かりませんが、オンライン版の部誌はカラー付きなので是非そちらもご覧ください(図③)。

(4)パーツ分け

(1)～(3)の工程で完成させた絵を Live2D のためにパーツを分けていきます(図④)。(2)で
レイヤーを整理したのがここで活きてきます。線画と塗りのレイヤーを統合したり微調整を
したりして立ち絵作りは終了です。立ち絵が完成したので次は Live2D を使っていきます。



図①



図②



図③

Live2D でキャラクターを動かす！

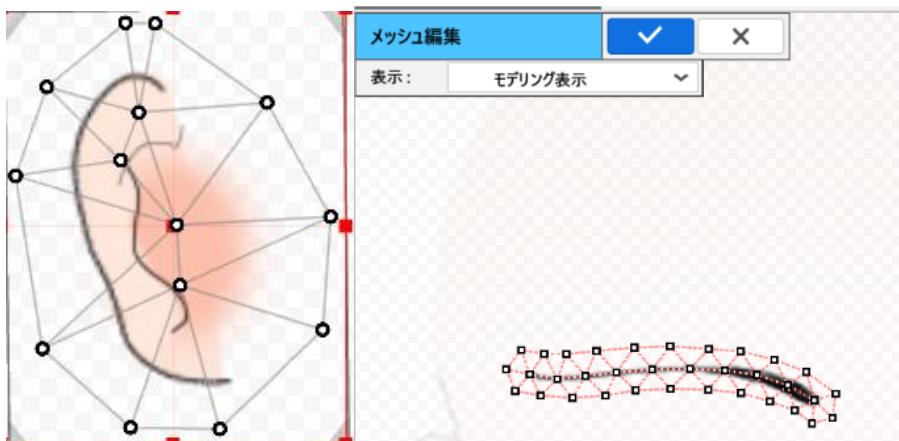


図④

②Live2D 内でのパートの下準備

Live2D でキャラクターを動かすためにはパートそのままではなくパートに「メッシュ」というものをつけなければなりません。

「メッシュ」というのはパートを動かすための骨組みのようなものです。左下図の小さな点の集まりがメッシュです。この点を動かすことによってパートを変形できます。(左下図は右耳) メッシュはある程度自動で生成することができますが、顔や目鼻などの細かいパートは手動でメッシュをついた方が綺麗に変形できます。



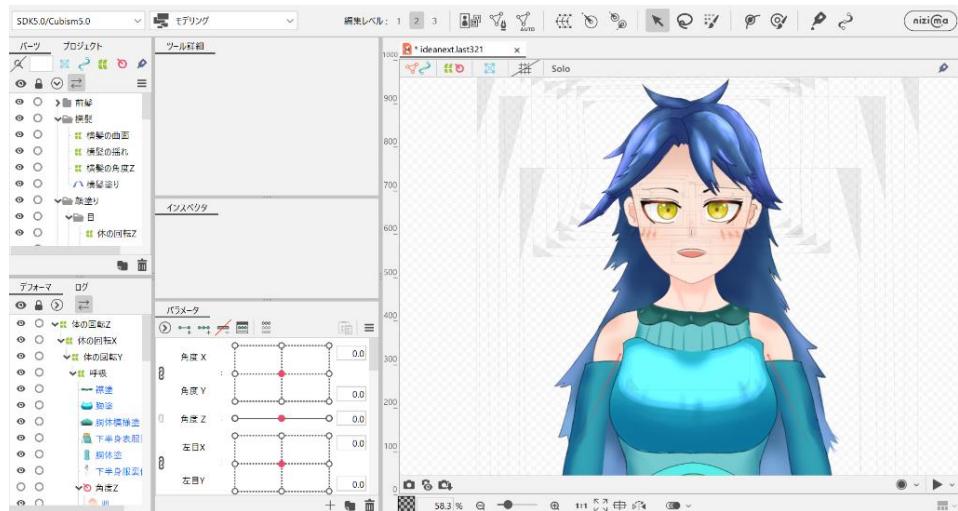
(図は右耳)

(右眉毛のメッシュを手動でついている図)



(左は手動、右は自動)

Live2D でキャラクターを動かす！



(Live2D モデリング作業画面)

③パラメーターの作成

②でメッシュを作成したのでようやくパートに動きをつけていきます。

動きをつける前にいくつかのパートをデフォーマというものにまとめます。

デフォーマはパートをまとめる箱のようなもので回転デフォーマとワープデフォーマがあります。回転デフォーマは文字通りパートを回転させワープデフォーマはパートの大きさを変えたり変形させたり位置を変えたりさせます。



(左 回転デフォーマ 右ワープデフォーマ)

それではパートとデフォーマごとにパラメーターをつけていきます。

パラメーターは何種類もありますが今回は頭を動かすところだけ説明します。

Live2D でキャラクターを動かす！

(1)角度 Z のパラメーター

主に首をかしげるときに使う動きです。パラメーター上のキーフォームに状態を対応させることで動きをつけられます。言葉だけでは分かりづらいと思いますので図で説明します。

(2)角度 X のパラメーター

顔が左右を向く時の動きです。(1)と同様に顔が右を向く時と左を向く時の状態をキーフォームに設定します。

(3)角度 Y のパラメーター

顔が上下を向く時の動きです。(1)、(2)と同様です。

(※ここでいう X, Y, Z はいわゆる Y-up という縦方向を Y 軸とする座標系のことです。)

キーフォームが真ん中



キーフォームが最小

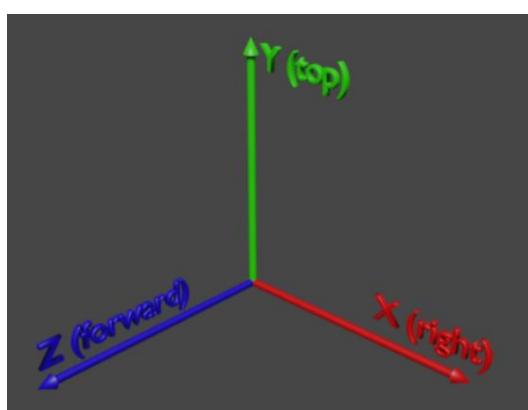


キーフォームが最大



上図のように状態を設定します。

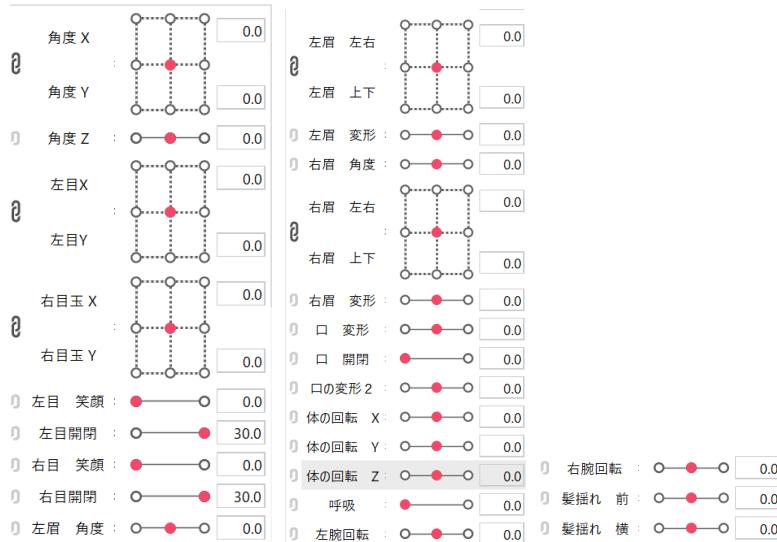
(↓これが Y-up)



上図のように状態を設定します。キーフォームの最小から最大までの動きは自動で補完されるので最小と最大を設定することで首をかしげる動きが作られます。

Live2D でキャラクターを動かす！

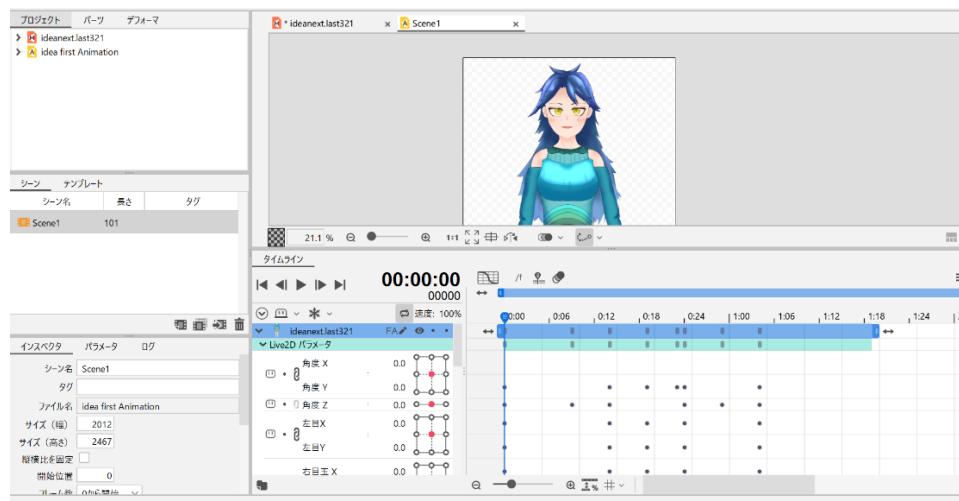
頭を動かすためのパラメーターだけではなく瞬きや口の開閉などのパラメーターもあります。以下が今回作ったパラメーターです。



④アニメーション作成

③でつけた動きを映像化していきます。

Live2D モデリングモードからアニメーションモードに切り替えると下図のような画面



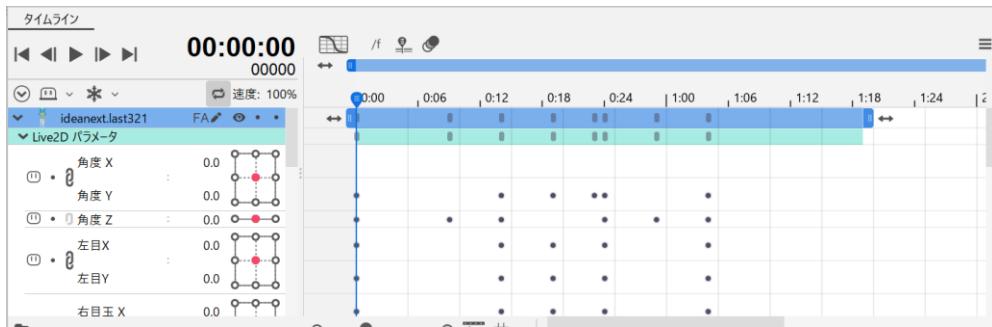
この画面のタイムラインにパラメーターを適応することでアニメーションが作れます。

要領としては③と同じでキーフレームによってアニメーション化されます。

タイムライン上のキーフォーム（前述のキーフォームとは

異なり、秒+フレームレートで表される数字のことです。）に状態を設定します。

Live2D でキャラクターを動かす！



あとは再生ボタンを押して出来栄えを確認し、動画として書き出せば動くキャラクターの完成です！！

これで今回の目標は達成ですが次の目標としてモデルを自分の動きに追従（トラッキング）させて動かすということに挑戦してみます。

部誌執筆時点 {3/25 午前1時（徹夜…）} では動画として書き出すまでしかできていないのですが、文化祭当日 {5/2, 5/3} にはもしかしたらモデルのトラッキングを展示しているかも

されません。されていて欲しい！！されていたらウレシイ！！

ということで文章は以上です。最後までご清聴下さりありがとうございました。

四輪オムニ機体の制御と制作秘話(?)

四輪オムニ機体の制御と制作秘話(?)

K. T.

(挨拶?パート)

自己紹介で締切り直前に BALATRO にハマってしまった、部誌提出どうしよう的なことを書きましたけれども、この天才的な会計様が部誌の提出をしない訳が無かろうなのだ！！！ということで四輪オムニ機体の制御アルゴリズムと制作秘話について話していこうと思います。

1. まず四輪オムニ機体って何？

そもそも四輪オムニ機体とは何かについてざっくり説明致しますと、オムニホイール（外円上にローラーを付けることであらゆる方向に移動ができるホイール、図 1）を円形に 4 つ配置することで、回転をせずに全方向に移動できるロボットのことです。この記事では私のチーム（私、部長、K 君、M 君）が加賀ロボレーブで出したセンサー 4 つの簡単なタイプの四輪オムニ機体（図 2）について説明していきます。

※CN○とはマイコンのチャンネル○につないでいるラインセンサーのことを指します。



図 1、オムニホイール

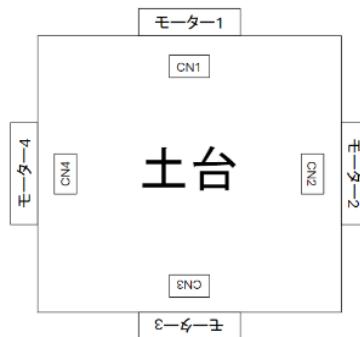


図 2、ざっくりした四輪オムニ機体

2. オムニ機体の制御について

先ほど言った通り、今回お話しする四輪オムニ機体はセンサーが 4 つしかないので、斜め移動ができません。なので、センサーが 12～個のタイプのように斜め移動するときに $\sin \theta$ と $\cos \theta$ の値を取得して…といった複雑な手順は必要ありません。やったね！！なので、基本の動きとしてはセンサーが反応している方向にモーターを動かすだけでいいです（図 3）。ですが、直線を走っているときに走っている時に反対側方向のセンサーが反応してしまい、正常に走らないといったことがあるので、対策しなければなりません。しかし、「どちらの方向に移動している時に」を直接制御するのは難しいので、センサーの反応と変数を連動させる方針で行きます。詳しく説明すると、CN1 が反応している時に変数(sensor)を 1 にし、sensor==1 の時は CN3 が反応しても sensor=1 のままになるようにします（図 4）。また、センサー反応から変数制御→モーター出力という工程にはどうしてもタイムラグが発生してしまうので、sensor の値が変化した時に逆方向に少し進むというプログラムを加えます（図 5）。

四輪オムニ機体の制御と制作秘話(?)

ライントレースでは通常 T 字路を正しい方向に曲がれるかというのが課題になってくるのですが、四輪オムニ機体は優先する移動方向を指定しやすいので、考える必要はありません。

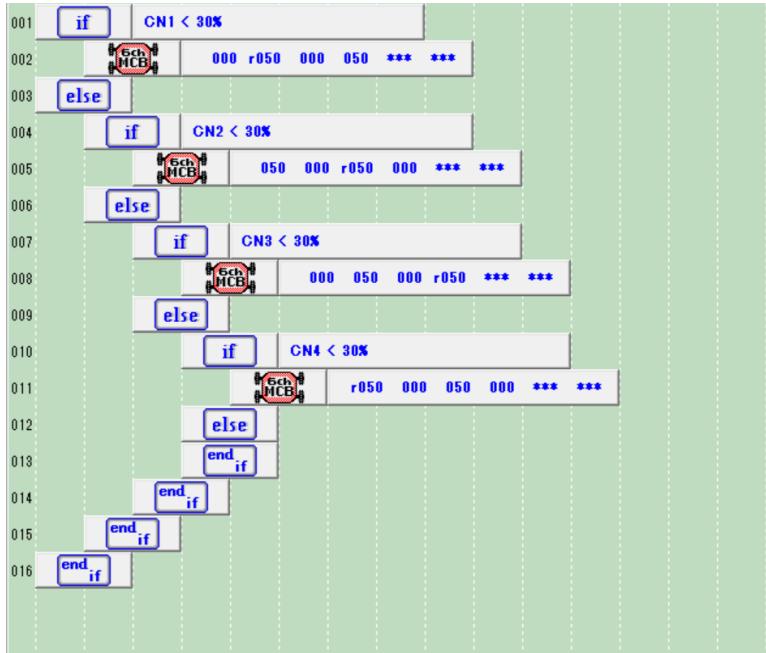


図 3、基本形



図 4、変数制御

四輪オムニ機体の制御と制作秘話(?)



図5、プログラムの完成形

3. 制作秘話(?)

ここでは主にこの四輪オムニ機体を出した加賀ロボレープでの話をします。大阪ロボレープでボロ負けした私たちはもっと前から準備をしておこうといっていたのですが、なんだかんだ準備が本格的に始まったのは大会二週間くらい前だっけな...まあなんにしろ準備を開始するのがギリギリだった気がします。こんなに準備が遅くなった言い訳としましては、オムニ四輪の土台にする予定だったボードの在庫がなく、届くのに非常に時間がかかったことがあります。私たちは大阪ロボレープが終わってから準備始めよう!!とか言ってたんですよ。そうですそうです。私たちは悪くありません。(結局大会では使う予定だった土台を使わず、ユニバーサルボードにモーターとセンサーとマイコンを付けて出したので判断が遅いだけな気もしますが...)まあなんやかんやあって大会一日目、加賀へと向かいました。加賀へと着いたのは12時頃だったので、駅の近くのショッピングモールのフードコートで昼食を取りました。私はここで担々麺を食べたのですが、この時私は知りませんでした。大会の三日間、コンビニ飯で同じような物ばかり食べることになり、定食を食べておけば良かったと後悔することになるとはまだ知りませんでした。一日目、これは現場調整期間なのですが、この時なんと私たちが作った四輪オムニ機体のセンサー間隔が大きすぎてコースに干渉してしまうことが発覚しました。まずいですよ!!!ここはセンサーの間隔を小さくし、なんとかなったのですが、部室での準備では出せて安定していたモーター出力が出せなくなってしまいました。

四輪オムニ機体の制御と制作秘話(?)

私たちとしては大会直前に機体の体積制限とラインの太さしか教えてもらっていないから、センサーの間隔がどれくらい以上であれば干渉してしまうということも教えて欲しかったです(普通のライントレースロボットのセンサー間の長さはそこまで広くないので我々のロボットが変なだけかもしれない)。練習の時のモーター出力が出せていれば予選通過も出来たかもしれません。センサーの間隔を狭め、一旦コースを走れるようになって一日目は終了。私たちとしては満足していたのですが、最大の難関、カゴに取り掛かってすらいませんでした。二日目、カゴの製作、設置に取り掛かりつつ、モーター出力を上げるために調整をしていました。中学の部のコースがいっぱいだったので、私たちは高校の部のコースで練習をしていました。高校の部のコースの方がラインが細く、T字路も2つあるのですが、なぜか中学の部のコースよりも高校の部のコースの方が上手く行ったので、高校の部で出場できないかななんて言っていました。そんなことはどうでも良くてですね、肝心のカゴが問題なんですよ。カゴの何が問題かと言いますと、まず重さがあるのでモーター出力が下がってしまう点です。カゴにはピンポン玉を入れて運ばなければならないので、ピンポン玉の重さも相まって車体の重量が上がってしまい、モーターがパワーを発揮できなくなります。もう一つは、重心が不安定になることです。カゴの固定が弱ければ、重心が揺れ、ガタガタして非常に不安定な走行になります。そこで我々は支柱を十字クロスにし、同じ高さの支柱の穴に竹ひごを通して、と色々工夫をしたのですが、結局重心が安定せず、脱線などが多くなりました。男はつらいよ。来年こそは… カゴしっかりしような(自戒)。三日目はひたすら測定ですが、測定をするチームが多すぎて最後まで測定できるか分かりませんでしたが何とか測定しきりました。しかし、結果は点数が足りずに予選敗退。つらい。

4. 余談

ここまで書いて、部誌の締切り直前に提出して、部誌編集の時にざっくり見直してみたら、なんとびっくりコンパスセンサーの存在を忘れていました。という訳でほんの少しだけコンパスセンサーのお話をします。コンパスセンサーとは、マイコンの電源を入れてからの傾きを測ることができるものです。四輪オムニ機体は、理論上は回転なしで動けるものなのですが、それはあくまで理論上の話です。やはり摩擦やモーターの回転数のずれというものは少なからず発生しますし、先ほど言った通りカゴがガタガタし、重心が不安定なために機体が多少回転してしまいます。そこでコンパスセンサーを使って、スイッチを入れてからの傾きが 5° 以上になったらもとに戻すというプログラムを加えていました。四輪オムニは回転が不要なだけで回転しようと思ったらできますのね。あと、2.のところでお話しした四輪オムニ機体はT字路を曲がりやすいということに関して少し説明いたしますと、if文は基本上から順番に実行されていきますので、例えばT字路がCN1の方向とCN3の方向に分岐しており、CN3の方向に曲がりたい時は、CN3が反応したときにsensorを3にするというif文をCN1が反応したときにsensorを1にするというif文よりも上に持っていくだけで良いのです。これが、四輪オムニ機体はT字路を曲がりやすい理由です。ここまで読んでいただき、ありがとうございました。

美少女の髪の毛作ってみた

美少女の髪の毛作ってみた

K. T.

他の部員が部誌を書かなさ過ぎて2個目の記事を書くことになった会計です。つらい。という訳で今回は、Blenderという3DCGソフトで美少女の髪の毛をモデリングしていきます。

1. 髪の毛の構造

本来髪の毛とは一本一本生えている物ですから、理想としては一本ずつ作っていきたいのですが、そんなことをすると労力と頂点数がとんでもないことになるので、一束の髪の毛を「房」として考えて作っていきます。基本アニメや3Dゲームのキャラクターは房から出来ています。私がよく参考にしているH○yoverseのキャラクターとかは顕著ですね。最近(?)見たこれの例外は推しの子のアイの死亡シーンくらいですかね。ほんとあのシーンの作画コストヤバそう。本題にもどると、右図1の髪の毛のグループを前髪、2のグループを触角、3を後ろ髪とします。前髪は主に顔のバランスを保つのに役立ちます。アニメチックのキャラクターの顔って結構バグって、顎～眼と眼～頭頂部の距離が同じ、何なら上側の方が長かつたりします。なので、髪の毛全部はぎ取った顔のモデルを見ると結構大変なことになってます。触角は、主に輪郭をごまかすために使われます。顔のモデリングが多少は適当でも誤魔化しようが効くんですが、その一つの要因がこれです。うれしいね!最後、後ろ髪はまあ髪型のアイデンティティとでも言いますかね、髪型の特徴が出る部分です。ある意味では一番大事。まあ、「房」と部位の話もしたところで、実際にモデリングしていきましょう。

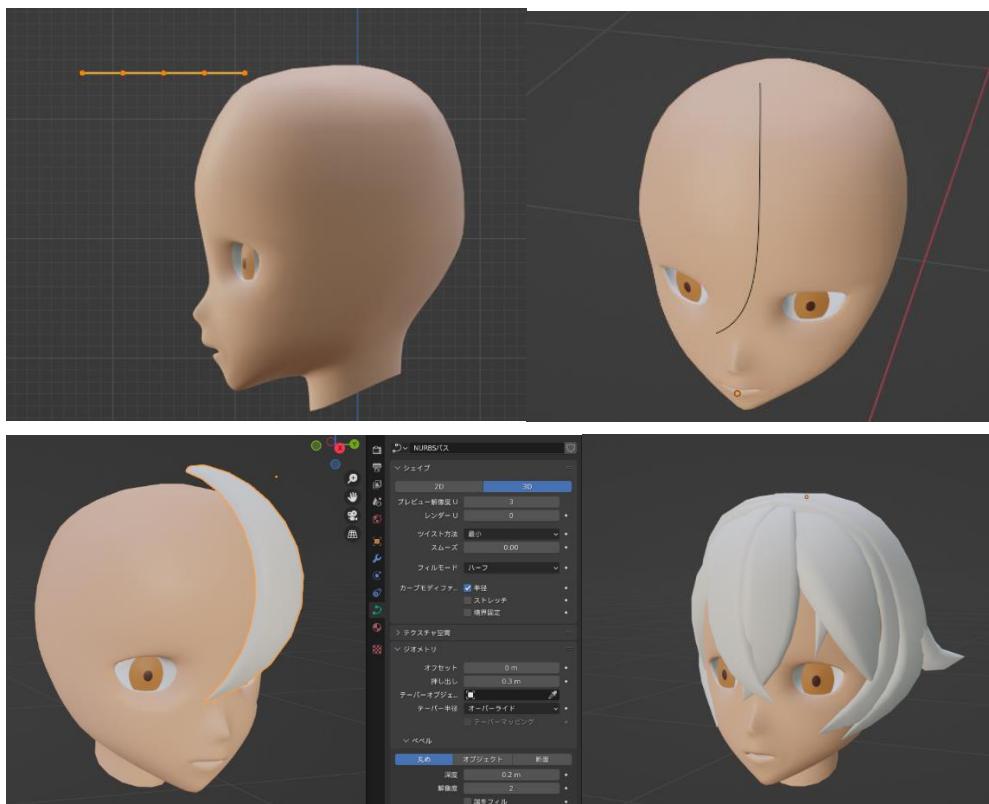


2. 実際にモデリングしてみた

髪の毛の房はバナナ状の物を作ればいいので、曲線を広げる方針で行きます。まずは体のモデルを用意しましょう。これはそこら辺のMMDモデルの服を剥ぎ取るなり自分で作るなりしましょう。作り方は調べたら結構出てくるので調べましょう。なぜか髪の毛の作り方はこのカーブ式があんまり出てこないんだよなあ...あと、私が見てたキャラクターモデリングの動画髪の毛編あたりから急にタイムラプスで解説無くなったりしたの許さんからな。Blenderの新規オブジェクト→カーブ→パスで左図のような線を呼び出します。そして、これを曲げて行きます。この頂点の一つ一つを動かすだけで自然な曲線を自分の思った通りに作れます。うれしい!これを右図のようにいい感じに曲げたら、押し出して行きます。

美少女の髪の毛作ってみた

左下図のようなオブジェクトデータタブを使い、「深度」と「押し出し」の数値をいい感じに調整し、顔に適合するように回転し、各点のまわりの大きさを変え、いい感じに回転させると左下図のようないい感じの房が作れます。この時、フィルモードをハーフにします。ポリゴン数が節約できるのと展開がキレイになります。また、プレビュー解像度 U と解像度は小さくしておきます。こここの頂点数が増えたところでそこまで見た目が変わるわけではないので。ポリゴン節約できるところは節約しておきましょう。重くなります。後はどんどん作っていくだけです。一旦前髪と触角だけ作ってみたのが右下図です。この後、細かい編集をしやすくする・マテリアルを作るとき、立体を平面に展開するためにオブジェクト>変換>メッシュでポリゴン化していきます。ポリゴン化したら髪の毛の房の先を枝分かれさせていきます。編集モードで枝分かれさせたい頂点を選択し、V キーで分離させていきます。そうすることで途中まで同じ房だけど先端が分かれている、といった髪の毛を房を増やさずに簡単に作ることができます。後ろ髪も同じように作っていきます。

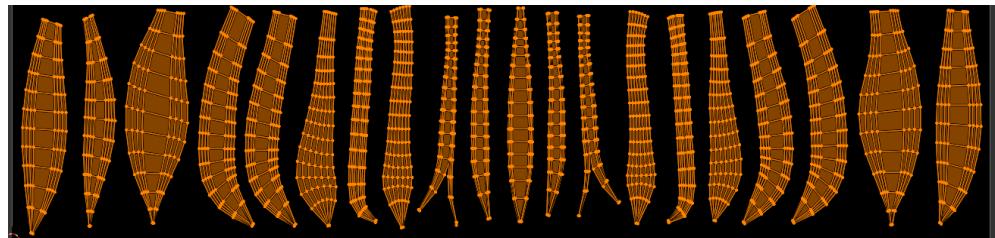


3. マテリアル(テクスチャペイント)

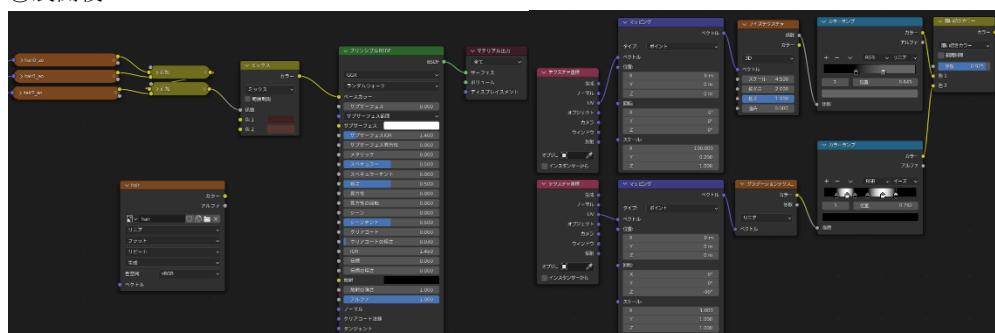
実は髪の毛を作るときのマテリアルってものすごく大切で、立体感や質感、影の再現、光沢等を出すのには欠かせない工程となっています。通常 blender で使うマテリアルはシェーダーを多用しますが、キャラクターの 3D モデルの形式としてよく使われる .pmx という

美少女の髪の毛作ってみた

拡張子ではシェーダーが適用されず画像テクスチャしか反映されません。なので、基本的にキャラクターの3Dモデルのマテリアルは画像テクスチャだけで作らなければなりません。このことを踏まえて、髪の毛のマテリアルを作っていくます。まずはオブジェクトの展開ですね。テクスチャ画像のどの部分が3Dモデルのどこに対応しているかを明確にするために展開していきます。すべての房を選択し、編集モードにした後、Uキーで展開していきます。この時、もし2つのカーブのフィルモードがフルになっていない場合、展開図が汚くなります。展開したら、この後の編集が楽になるように各房の展開図を並べ替えていきます(①)。並べ替えたら、まずは髪の毛の陰影をはっきりさせるためにアンビエントオクルージョン(AO)を適用させます。髪の毛を3~5個に分けた後、各部位ごとに無地の画像を用意します(②)。用意したら、そこにベイクタイプをアンビエントオクルージョンにしてベイクしていきます(④)。ベイクしたら、そのすべての画像を加算した後、ミックスで白黒→自分の設定したい髪の色・影の色にします。こうすることで、髪の毛の房と房の間をはっきりさせ、凹凸感を出しています。できたら、このマテリアルを一つの画像にベイクしていきます。ベイクタイプをディヒューズにして、最終的に使用する画像をよういして、ベイクしていきます。ベイクできたのが右上図です。アンビエントオクルージョンで凹凸感が出せたら、次はリアルな質感を出していくます。ノイズテクスチャで縦の縞を作り、髪の毛のボリュームを演出していくます(⑤)。できたら、次はグラデーションテクスチャとノイズテクスチャを合わせて光沢感を出していくます(③、⑥)。ノードが組めたら、それをテクスチャ画像にベイクしていきます(⑦)。これで暗影の画像テクスチャと質感、光沢の画像テクスチャが完成したので、それを合わせていきます(⑧)。完成したのがこちら(⑨)です。

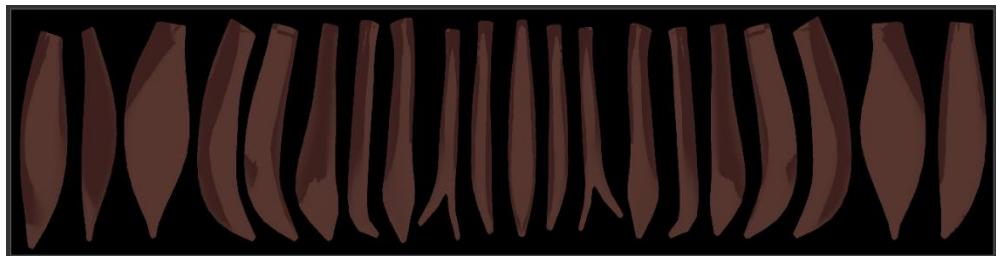


①展開後

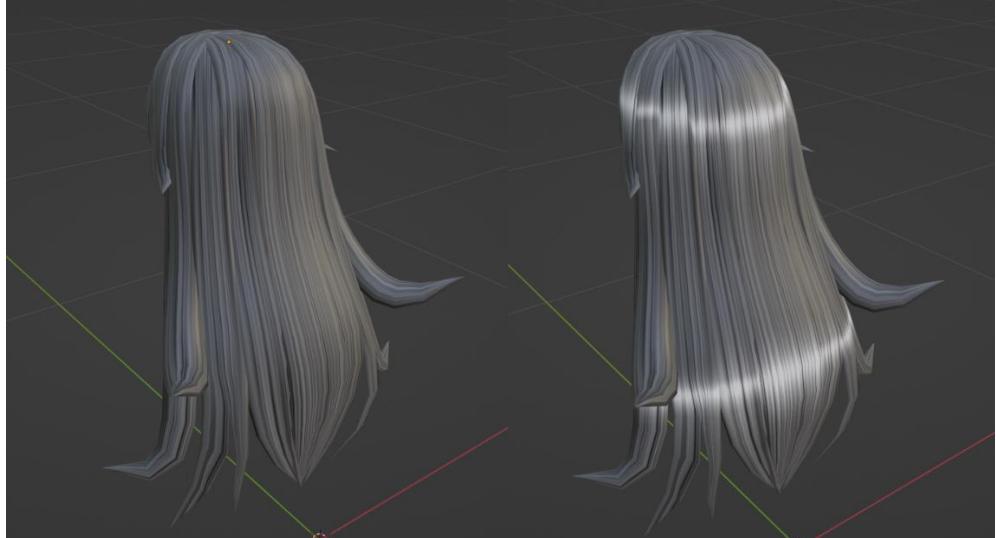


② AO をベイクした後、それを合わせるノード ③ 髪の光沢、ボリュームを出すノード

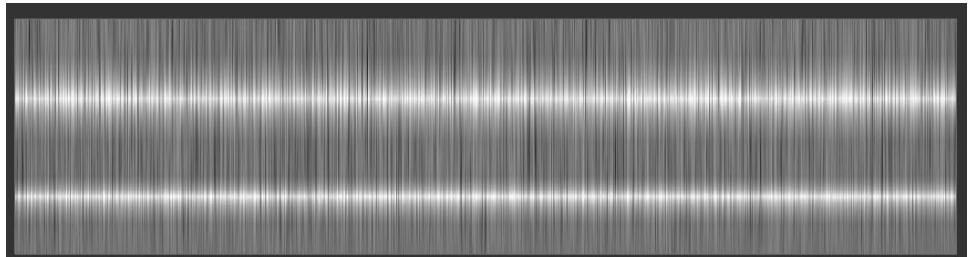
美少女の髪の毛作ってみた



④ AO をベイクした結果の画像

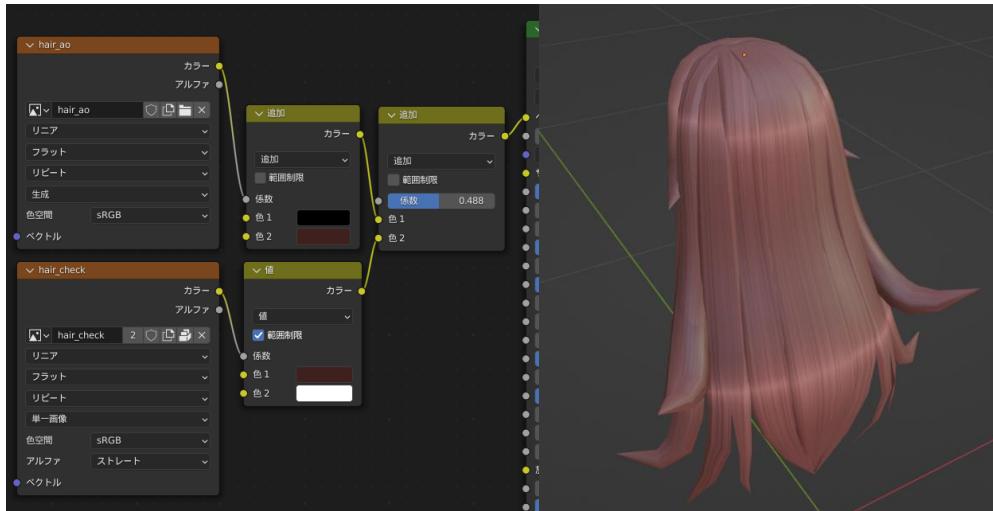


⑤ 髮のボリュームだけ(ノイズテクスチャのみ)⑥ 光沢を加えたもの(③の画像のノード)



⑦ ③のノードをベイクしたもの

美少女の髪の毛作ってみた



⑧ ④と⑦の画像テクスチャを合体させるノード ⑨ 完成形

4. 余談

栗原「もしもし雨穴さん、先ほどいただいたモデル…いや…これ、かなり不可解ですよ。」
栗原「ここ…分かりますかね…2. のモデルを横から見た画像、これ、blender 標準の四角(大きさ 0.5m)に比べたら顔がかなり大きいことが推測されるんですよ」栗原「ここからは私の推理にはなってしまうんですけど、恐らくこのモデルの大きさは、クソデカイ力士の可能性が…」雨穴「そうですか、じゃあ、これも大丈夫そうだと伝えておきます。」しばらくして…
雨穴「栗原アアアアア!!!耳ねえぞオオオオオ!!!」はい、この私が用意したモデル、耳を作るのを忘れていました。あと単純に顔が 1.5m くらいとクソデカイです。やらかした。あと、実際に作りながらここまで書いてよくよくみたら AO 適用してもしてなくてもそこまで変わらないなと思いました。やはり AO というのは複数のオブジェクトが重なり合っているところで使用したり髪の毛の内側が見えるときにしたりするものなので今回の髪型を作るうえでは不要なんですね。あと、今回はノードを複数組み合わせて作りましたが、本来テクスチャペイントってブラシを使って作るもんなんですね。まあその場合板タブと画力が必要になるので、両方とも持っていない私は今回ノードを画像にペイントする形にしました。あと、マテリアルをよりリアルにするにはバンプマッピングを適用させてもいいかもですね。さて、ここまで読んでくれた人は読んでくださりありがとうございました。適当に飛ばし読みをしてここまでたどり着いた人もありがとうございました。気が向いたら実際に blender を触りながら読んでみるのも良いかもですね。以上です。

はじめに

こんにちは。80回生の「よつばのくろおばあ」です。自己紹介に「よつば」「みつば」「ふたば」が出てきて「何言ってんだこいつ」となったと思うので一応補足しておくと、全部SNSのアカウント名です。「よつば」が僕の本垢で、「みつば」がサブ垢。「ふたば」は友達が僕の真似をしてふざけて作った垢です。だから「たまに『ふたばのくろおばあ』になります」とか自己紹介で書いたけどほとんど「ふたば」になることはありません。まあ所詮ふたばとか葉っぱ2枚しかないクソ雑魚なんで。(意味不明すぎるマウント)

題名を見て「Blenderの使い方を教えてくれるのかな?」と思ったそのあなた。残念ながら、僕はBlenderの使い方をきちんと勉強するほど真面目ではないので、期待したようなクオリティの解説はなされていない可能性が高いです。極度の面倒くさがりが適当にBlenderをいじった結果を綴った読み物として眺めることをおすすめします。

そういえば去年の部誌で「来年は今年の続編書くよ」みたいなことを言った気がするのですが、たぶん気のせいですよね。PCの奥底から引っ張り出したところ、「プログラミングとかに詳しくなったらソフトウェアに関する続編を書くかも?」的な記述がありました。あれから1年、一切プログラミングの勉強をせずに放置していたくせに、この前情報オリンピックに出場した友達に「俺も情報やろっかなw」みたいな発言をしてしまいました。まあ僕なので仕方ない。というかなんか去年の部誌の方が全体的に真面目ですね。考察とか書いてる。昔の俺偉い。

それにしても、僕が理由もなくこんなに長々と前置きを書くはずがありません。もちろん本編を書くのが面倒くさいので字数稼ぎをしたいからです。まあこういうゆる~い感じの文章を書くのは嫌いではないんですが。

イントロダクション

「はじめに」と「イントロダクション」ってほぼ同じ意味だよな。まあいいか。

生徒会関係の仕事でCGアニメーションを使いたくなったので、BlenderというCG制作アプリをインストールしてみました。開いてみたはいいものの使い方がよく分からず放置していたのですが、そうこうしているうちに部誌の1次〆切が迫ってきました。そこで、せっかくならBlenderの超基本的な使い方をそれっぽく部誌にまとめてみようと思います。本とか買ってもいいんですけど、僕の場合どうせ3日後には本棚に収まってるのあんまり意味ないです。あ、ちなみに現在時刻は生徒会の仕事の〆切の6時間前です。持つべきは働いてくれる友達ですね。

適当すぎる Blender 使い方講座

インストール

こういう専門系のアプリって地味にインストールが難しいやつが多いんですけど、Blender は適当にネットで調べたらインストールできました。ホームページとかもわりとしっかりしてるのでやりやすかったです。ちなみに TEX っていうプログラミング的な感じで文書を綺麗に作成するというアプリがあるんですけど、TEX はいまだにインストールできません。

まずは開いてみる

とりあえず開いてみました。空間の座標平面が広がっていますね。右上の x 軸とか y 軸の表示で視点の角度を操作できるようです。その下に 4 つくらいアイコンがあるのでこちらも押してみると、特定の視点に変更する機能だと分かりました。こういう機能は作業を効率化するための機能なので、使いたくなったら使います。使えなくても多分なんとかなるっしょ。次は真ん中の立方体を見ます。原点中心、1 辺は方眼 2 マス分といったところでしょうか。右に目を向けると Cube というのがあるので選択してみます。右下で詳細情報を確認できるみたいですね。こういうのは創作アプリあるあるなので、トランسفォームとかの数値をいろいろ変えてみて、挙動を確かめるとか言いつつ遊びましょう。同じ要領で Light と Camera にも目を向けてみます。Light は空中に浮かんでいる丸のことですね、影の問題に登場しては受験生を困らせてくる点光源に似たデザインをしています。Camera って角度 0° にすると真下(z 軸負の方向)を向くんですね。おもしろ。

物体を増やす

立方体 1 個では遊ぶのには限界があるので、物体を増やしてみたくなりました。物体を増やす方法を模索してみると、左端にありました。+ が立方体の左肩に載っているアイコンですね。アイコンの右下に切れ込みが入っています。これも創作アプリあるある、長押しすると拡張機能が展開されます。適当に円柱とか選択してみましょう。フィールド上で軽くドラッグしてみると円柱が出現します、が、楕円柱になってしましました。楕円柱の各種数値を編集して、正円柱に補正しましょう。スケールの数値を変更すれば大きさが変わるのでそこを編集…と思ったのですが、スケールの数値が全部 1.000 になっています。どうやらスケールというのはいったん作った物体をどのように引き伸ばすかということらしいですね。楕円は適当なドラッグで作ったので、どんな比率で引き伸ばせば正円になるのか分かりません。困りましたね。しかし、筆者は極度の面倒くさがりです。毎回 Google を開いて「Blender 正円柱 作り方」とか調べて信用できる検索結果を精査するなんて選択肢はありません。ということで、例によって創作アプリの勘に頼ります。Word や Excel の類でも、円を作図するときは Shift を押しながらドラッグすると正円になるので Shift を押しながらやってみます。予想通り正円柱になりました。高さも直径と同じ長さになったようですね。ここからスケールの値を変更すれば、自分のほしい比率の円柱が作れます、が、比率が自在になっただけで大きさは自在ではないですね。どうすれば大きさも自在になるのかといろいろ試してみたところ、Ctrl キーを押しながら選択すると既存の頂点の座標を選択できることに気付きました。

適当すぎる Blender 使い方講座

ゲームセットですね。初期設定で配置されている立方体の頂点に合わせて円柱を作れば、半径 1 マスの正円を底面とする円柱が作れます。円錐も同じ方法で攻略できました。

球は 2 種類ありますね。日焼け止めとかでよく見る名前と日本版の会長がやらかした委員会に似た名前ですが、よく分からないのでとりあえず両方作ってみます。紫外線の方は地球儀みたいな見た目ですね。あとは…そうですね、虫の卵にも似てますね。ちなみに何かを虫の卵に例えると嫌がられるので皆さん気を付けましょう。委員会の方は正 20 面体風の謎立体です。とりあえず紫外線の方が図形的にも世間的にも無難なので、ひとまずは紫外線を使うことにします。

ブレイクタイム

1 次〆切に間に合わないことを悟ったので球を多少いじったところで筆を置きました。その後しばらく PC の奥底に眠っていたのですが、2 次〆切が近いので引っ張り出してきました。まあ 2 次〆切が近いと言っても「もうすぐ〆切が来る」じゃなくて「このあいだ〆切が来た」なんですね。ちょうど〆切のあたりに予定が重なったから書く時間がなかった、と言いくつておきます。今日の 24 時までに提出したらセーフらしいのでまあいいや。

せっかくブレイクタイムと銘打ったので、なんか雑談でもしますか。まじでアマ研とは関係ない話をするので興味ない人は読み飛ばしてください。あと文化祭 YouTube のネタバレを含むのでご注意ください。僕アマ研以外にディベートをやってるんですけど、今年の文化祭 YouTube 企画でディベート同好会が一般人とディベートしてみた、的な動画があって、ディベート同好会の友人がそこでボロ負けしたらしいんですよね。その友達を煽りたくなったのでその議題について考えてみたいと思います。

論題は「ペットにするなら犬かダニか」というもので、ディベート同好会はダニ派で不利だけど流石に一般人には勝てるよね、みたいな企画とのこと。とりあえずダニ派として自分が喋りそうなことを気の向くままに書いてみます。

犬とダニどっち飼うかっていう話をすると現実的に一番考えないといけないこととして、コストの問題がある訳ですよ。犬ってドッグフードとか動物病院代とかめっちゃ金かかると思うんですけど、ダニって無料で飼えるんですよね。てことはまずこの時点でダニの方が有利。(中略)で、ダニを飼う目的なんですけど、布団ってあるじゃないですか。あれ干すと結構いい匂いすると思うんですけど、これって実はダニのおかげなんですよね。で、嗅覚的な幸福を得る方法って、現在の科学でもってしても芳香剤ぐらいしかないんですよね。芳香剤って高いじゃないですか。その点で、無料でいい匂いを提供してくれるダニの効果っていうのが分かる。相手側さんはたぶん主にダニが体に悪いみたいな話をくると思うんですけど、人間って、適度に汚い環境に置かれる方が免疫力 UP して体にいいんですよね。なので、まあダニが健康に与える影響っていうのは、むしろこちら側に有利なんじゃないかなーと思いますけど。

終わり。なんか読み返すとめっちゃ腹立つ書き方をしてますね。心なしかひろ〇きに似てる。ちなみに布団の匂いはダニというよりダニの死骸らしいです。

適当すぎる Blender 使い方講座

実際の動画を見せてもらったんですけど、なんか一般人とか言いながらめっちゃ喋りが上手かったので、僕が出演してもたぶん負けます。勝てるとしたら健康被害のターンアラウンドの話とかかなー。ターンアラウンドはディベートの専門用語で、相手の主張を逆に利用して自分側の味方につけることを言います。まあ犬 vs ダニ以外にもう 1 戦撮影したらしいので、気になる人は YouTube で動画を視聴してみてくださいね。あと、僕の友達を煽りたい人はボクシングという文化祭ディベート企画を見てみよう。昼くらいに大講義ステージでやってます。よし、宣伝完了。

画像とか映像とか

さて、Camera は CG 映像を作るときに使うみたいですが、CG 映像を作る話は面倒くさそうなのでまた別の機会にします。しかし完全シカトするのも可哀想なので、下絵というのを使ってみましょう。PC 内に保存してある画像を追加するみたいです。なんかおもろい画像ないかなー。某教師のコラ画像があったので取り込んでみましょう。嘘です。僕はコラ画像を保存するような人間ではありません。コラ画像を作るタイプの人間です。嘘です。実際の僕はコラ画像を作った経験はないです。画像編集は好きだけど、コラ画像はあんまり作ろうとは思わないかな。

特に変化はないですね。つまんな。ということで最初に視点をいじったときにカメラ視点があつたことを思い出してカメラ視点にしてみます。画像が表示されました。成功っぽい。追加した画像が背景になって映っています。どうやらカメラの視界に合わせて画像の縦横比が変わるものですね。背景より上に立方体とかが見えているので、先生の肖像とか Blender で作って背景の上に配置するとマジでコラ画像になりそうですね。コラ画像好きの友達に教えてあげよう。ダメか。

光源

あとはLightの話くらい軽くしておきましょう。もっといろいろ話題はありそうなのですが、CG 映像とか撮影絡みの話が多そだしお、シンプルになんか難しそうなので放置します。放置するほど強くなる。あつそれ少女だけか。

Light にもいろいろ種類があるみたいで、種類をいろいろ変えると光が当たっている範囲も変わるようにです。ただ光の当たっている様子が観察しにくいので、どこか調整できるところはないのかということでもまたもやアプリ内を探します。それっぽいのがありました。画面右上のモード切り替えみたいなやつで表示の仕方を変更できるようです。影が見やすいようなモードに変更します。いい感じ。光の種類や対象となる立体をいろいろ変えると、種類ごとや立体ごとに結構違いがあつたりしておもしろいです。CG 映像を作るときは光源の位置を場面に応じて変更したりするとかっこよさそう。いつか光源を自由自在に操れる人になりたい。こんな風に書くと電気タイプの能力者みたいですね。

適當すぎる Blender 使い方講座

おわりに

本格的に眠くなってきたのでシメにかかります。この部誌の唯一の学びは「標準の動作スタイル(創作アプリあるある)を知っておくと初見のアプリでもなんとかなる」でしょうか。Blender に興味を持った人は PC なり何なりにインストールして、僕より真面目に勉強すると楽しくなってくると思います。知らんけど。僕は楽しむ段階まで進んでいないので保証しません。

来年の部誌では Blender の機能についてもう少し詳細に執筆するだろう、とフラグを立てておきます。実際はプログラミングとかの話を書いてるかもしれません。一つだけ言えることは、どんなテーマで書こうと〆切には間に合わない、ということです。灘校生の悪いところですね。

現在時刻 26 時。2 時間過ぎちゃったけど大丈夫っしょ。提出用フォームを眺めていたら先輩たちが賢そうな記事を書いててビビったんですが、気にしないことにして就寝します。

ここまで読んで頂きありがとうございました。楽しんで頂けたら幸いです。おやすみ。

帝国海軍の電波探信儀

帝国海軍の電波探信儀

眠い人

どうも、眠い人です。

まず初めに、私は電波だなんだといったことに関してほとんど知識がなく、この文章は短期間で調べた内容で書いています。

なので、間違っているところなどもたくさんあるでしょうが温かい目で見守っていただけたらと思います。（これでいいんでしょうか）

今回、私は帝国海軍の電波探信儀（レーダー）について調べました。

きっかけは私がハマっているゲームです。

そのゲームの中で「21号対空電探」などが出てきたからです。

もともと、電探という言葉すら知らなかつたのでとても気になり調べたのが始まりです。

ただ、まだ細かいことどころか基本的なことさえあまり分かっていないので、

一旦は興味を持った帝国海軍の電探の系譜について調べてみました。

ちなみに、この時点では私がレーダーについて知っていたのは、電波が物に反射して戻ってくるまでの時間から距離などを測っているという程度の事でした。

まず、帝国海軍においては、レーダーの事を電波探信儀（略して「電探」とも）と言い、レーダーの発する電波を探知する装置の事を「電波探知機」と言っていたそうです。

そして、電波探信儀自体も一号から六号まであり、

一号 → 陸上での見張り・警戒用

二号 → 艦艇に搭載しての見張り・警戒用

三号 → 艦艇に搭載しての水上の相手を狙う用

四号 → 陸上で空中の相手を狙う用

五号 → 航空機に載せる用

六号 → 陸上で誘導を行う用

といった分け方がなされていたそうです。

他にも航空機の空一号から空十号までの区分などと、いろいろな区分があるそうですが今回は省略させていただきます。

帝国海軍の電波探信儀

例としては

一号、つまり陸上での運用のはずが小型軽量であることなどから艦艇にも搭載された「三式一号電波探信儀三型」(13号電探とも)や、

二号、つまり艦艇での見張り用でありながら最終的に潜水艦の潜望鏡(こんなに小さいものを見つけられるのかと驚きです。)などを発見できる性能を持ち水上の相手を狙うためにもつかわれることとなった「仮称二号電波探信儀二型」(22号電探とも)などがあるそうです。

まず、日本の電探開発は、1925年に帝国海軍が電離層の電波による高度測定研究が始まっていたそうで、そういう意味では日本海軍から始まったともいえるそうです。

また、1928年ごろには東京帝国大学でレーダーに関する研究が行われていたようで、第二次世界大戦より10年以上前からレーダーの基本となる無線技術は開発されていたようです

ただ、帝国海軍は当初、

「敵艦を探知するのに自分で電波を発射するのは恰も暗夜に物を探すのに提灯を用うる如きものである。

物を探し当てることは出来るかもしれないが、その前に自分の所在を暴露するものである。隠密行動を必要とする海軍に於いては必要のないものだ。」と、簡単に言うなら「電波を出したら見つける前に相手に見つかる」と興味を示さなかったそうです。

ただ、第二次世界大戦がはじまってからドイツなどでレーダーの有効性が証明され、開発が加速しました。

初期の電探は、たとえば陸軍の甲型電波警戒機であれば、送信機と受信機の間をいつ通ったのかが分かる程度のものでした。

これは1939年に開発に成功したものです。

それから2年後の1941年に開発が開始された陸軍の乙型電波警戒機は、その年の秋には開発が完了し、その性能はたった1年前のイギリスのレーダーと同じレベルのものだったそうです

さらに、乙型電波警戒機は1943年には車載型(車両に乗せたバージョン)、1944年にはさらに小型なものの開発に成功しています。

それでは少し海軍の電探について紹介します。

一号電波探信儀一型は陸上において空を監視する電探です。

これ以前の電探の三号電波探信儀(三号電波探信儀1型～3型などとは違うものだそうです。)などは発信機と受信機を離して設置して何かが間を通ったときに反応するというものでした。

帝国海軍の電波探信儀

それに対してこの電探は電波が対象に反射して帰ってくるまでの時間を計測するため距離を測ることができるというものでした。

性能に関してですが、単機に対して 70km の距離での探知に成功しています。

また、1 号機の実験から 1 年と少し経ったときには改良型の一號電波探信儀一型改一では 130km まで性能が向上しています。

これは最終的に百数十台が製造されています。

一号電波探信儀二型は一号一型の小型軽量版です。

重量は一号一型の 8.7t から 6t まで軽量化されましたが、単機で 50km の距離での探知となるなど性能面での低下が見られます。

一号電波探信儀三型は軽量化してもなお重く移動が困難であった一号二型をさらに軽量化しました。

その重量は分解すれば人力での運搬が可能な 110kg であり、その軽さから駆逐艦や巡洋艦に始まり果ては潜水艦にまで搭載されることとなりました。

性能は単機を 50km、編隊を 100km の距離で探知できるとしていました。

しかし、実際にはそれ以上の探知距離を誇り、空母「瑞鶴」において 242km の距離での探知に成功したとの記録も残っています。

二号電波探信儀一型は艦艇において空中の目標を探知する電探です。

性能に関してですが、単機に対して 70km、編隊に対して 100km の距離での探知に成功しています。

また、戦艦「伊勢」から戦艦「日向」を 20km の距離で探知することにも成功しています。

重量は 840kg で、戦艦や空母をはじめとした多くの艦艇に搭載されました。

ただ、一号電波探知機三型の実用化により主力が移りました。

それでも、すでに搭載されている分の多くは整備が続けられました。

二号電波探信儀二型は艦艇において水上の目標を探知する電探です。

性能に関してですが、1943 年 7 月中旬に戦艦「大和」に搭載して試験した際、戦艦を 35km、駆逐艦を 16km、潜水艦の潜望鏡を 5km で探知しています。

この電探には改四までの種類が存在し、最終的にはこの電探によるレーダー射撃で潜水艦に弾をあてるほどの精度を持ちました。

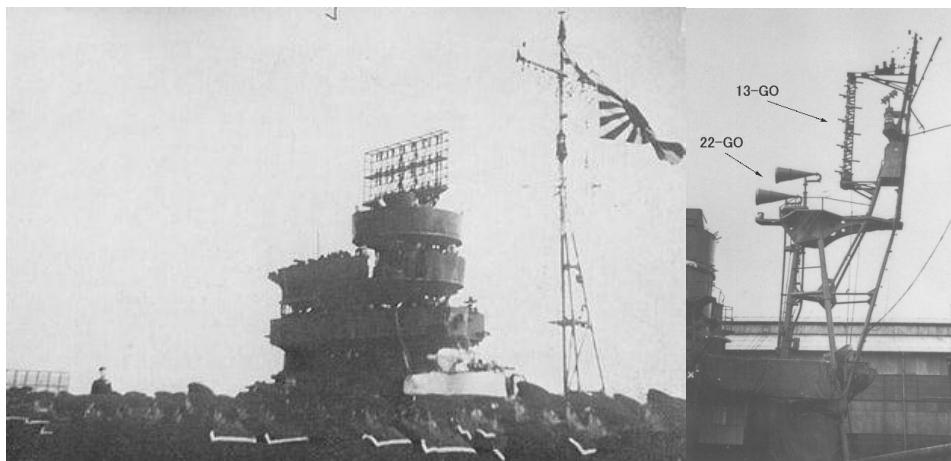
重量は 1320kg、潜水艦用であれば 2140kg で最終的に 2000 台ほどが製造されました。

他にも完成が遅れた結果艦艇への搭載がなされなかった二号電波探信儀三型や三号電波探信儀一型から三型といったものや、

台湾や日本本土の防空に用いられた四号電波探信儀一型から三型、

帝国海軍の電波探信儀

他にも終戦間際に完成し訓練中のまま終戦を迎えた、航空機が味方航空機の誘導に用いるはずだった六号電波探信儀一型・二型などがあります。



左から順に

空母「瑞鶴」の艦橋上部に搭載された二号電波探信儀一型、

駆逐艦「春月」のマストに搭載された二号電波探信儀二型と一号電波探信儀三型

最後に、今回私はこれらの事を調べて、より一層これら電探や無線関係の話に興味がわきました。

今後はもっと無線のことなどを知りたいと思っています。

来年の部誌にはもっとちゃんとしたことが書けたらなと思っています。

ここまでお読みいただきありがとうございました。

それではまた来年の部誌で。

帝国海軍の電波探信儀

参考文献・その他

国立公文書館アジア歴史資料センター

「昭和19年10月20日～昭和19年10月28日 捷号作戦戦闘詳報（比島方面決戦）
(3)」

<https://www.jacar.archives.go.jp/das/image/C08030036800>

Youtube

「日本海軍が対空レーダーで射撃指揮を行っていた！極限の状況化でのレーダー開発！技術者の数々の困難」

<https://www.youtube.com/watch?v=VyoBcZ3c2L4&list=LL&index=9>

「日本陸軍が本土防空用に開発したレーダーシステム【兵器解説】」

<https://www.youtube.com/watch?v=e-0HfFlomvg&list=LL&index=8>

日本帝国陸海軍電探開発史

「海軍電探開発史 本文 01」

<http://minoutai7.livedoor.blog/archives/18022252.html>

「海軍電探開発史 本文 02」

<http://minoutai7.livedoor.blog/archives/18022253.html>

Wikipedia

「電波探信儀」

<https://ja.wikipedia.org/wiki/%E9%9B%BB%E6%B3%A2%E6%8E%A2%E4%BF%A1%E5%84%80#%E9%96%8B%E7%99%BA%E3%81%AE%E7%B5%8C%E7%B7%AF>

「三式一号電波探信儀三型」

<https://ja.wikipedia.org/wiki/%E4%B8%89%E5%BC%8F%E4%B8%80%E5%8F%B7%E9%9B%BB%E6%B3%A2%E6%8E%A2%E4%BF%A1%E5%84%80%E4%B8%89%E5%9E%8B>

「二式二号電波探信儀一型」

<https://ja.wikipedia.org/wiki/%E4%BA%8C%E5%BC%8F%E4%BA%8C%E5%8F%B7%E9%9B%BB%E6%B3%A2%E6%8E%A2%E4%BF%A1%E5%84%80%E4%B8%80%E5%9E%8B>

「仮称二号電波探信儀二型」

<https://ja.wikipedia.org/wiki/%E4%BB%AE%E7%A7%B0%E4%BA%8C%E5%8F%B7%E9%9B%BB%E6%B3%A2%E6%8E%A2%E4%BF%A1%E5%84%80%E4%BA%8C%E5%9E%8B>

ARDFについて

k. k.

アマチュア無線研究部の活動内容に ARDF という競技があります。この競技がどのようなものかを説明していきます。

1. ARDF とは

ざっくり言うと森や山の中に電波を発信する装置 (=TX) が 5 個設置してありそれを受信機を用いて見つけるという内容です。使用する周波数は 3.5Mhz や 144Mhz です。国内のメーカーから受信機や TX は発売されていないので、中華製やオーストラリア製、チェコ製を用いる必要があります。アマチュア無線研究部では中華製の受信機を使用しています。TX はモールス信号を送信しており、聞こえてくるモールス信号 (※1) より TX の番号を決定しています。常に電波を出しているのではなく TX により出しているタイミングが違い、周期になっています。探したい TX が電波を出してない時にどのように進むのかも重要です。

※1 モールス信号と聞くと難しいイメージを持たれる方が多いと思いますが、これに関してはものすごく簡単です。TX1 は —— ··· (MOE) TX2 は —— ··· (MOI) のように最後の短点でどの TX かを判別できます。

② 実際の競技の流れ

ARDF は参加者がグループに分かれ、グループごとに時間をずらして出発します。競技時間は大会によりますが一時間から二時間ほどです (※夏の高校生全国大会では熱中症予防のため時間が短縮されました)。競技開始の直前に主催者から地図 (※2) が配られます。基本的に田舎で行うので地図には田んぼや山、送電線(←ここ重要です)などの情報が記載されています。

まず、搜索開始地点に着いたらイヤホンをつけて(それより前にイヤホンをつけると不正と認識されます)音が聞こえる (大きい) 方に向かいます。しかし山などがあると電波が反射して必ずしも音が聞こえる方向にあるとは限りませんので移動しながら定期的に方向を確認します。また、コンパスの所持が認められているためコンパスと地図を照らし合わせ自分の位置を把握し、自分の位置やすでに見つけた TX を筆記用具で地図に書き込むことで探索がしやすくなります。TX はどの順番で見つけても問題はありません。

このようにして TX を見つけていき、すべて見つけ終わっていなくても制限時間の 30 分ほど前になったらゴールの方向に向かうようにします。 (※3) ゴールは地図上に書いてあり、ゴール付近のゴールビーコンからも TX とは別の周波数で電波が常に送信されていますのでこれを頼りにゴールへ向かうこともできます。これが大まかな競技の流れです。

(※3) このようにするのは理由があります。TX は一つでも見つければ成績になりますが、制限時間を超過してしまうと見つけた TX の数とは関係なく失格になってしまうからです。

また、見つけた TX の数が同じ場合は早くゴールしたほうが上位になります。

③ TX 探索上の注意点

田んぼが多くありますので道に迷ってしまうこともあるでしょう、そんな時は先程重要と書きました送電線や給水ポイントを頼りに現在位置を把握するのがよいでしょう。

ARDFについて

各TXにたどり着いたことを証明するためにSIカードという小型のICチップを指に装着します。このSIカードをSIステーションという機械に差し込むと成績が記録されるわけですが、差し込む時間が短かったりすると正しく成績が記録されない可能性がありますのでピップという電子音とLEDが光ったことを確認しましょう。また、紛失する可能性がありますので腕時計などとゴム紐で繋げておくと無くすことはないでしょう（無くしてしまうと実費で請求されます）

④まとめ

以上の通りARDFというのは頭も体も使う素晴らしい競技です。始めるに際して受信機があれば簡単に始められます。（アマチュア無線研究部では受信機を部員に貸し出しています）みなさんも始めてみてはいかがでしょうか。

⑤資料

昨年度の夏、新潟で行われたARDF競技大会で使用した実際の地図です。また、資料提供にご協力いただきましたJARL新潟県支部・信越地方本部ARDF委員の佐藤久先生に感謝申し上げます。

第18回全国高等学校ARDF競技大会 @新潟県阿賀野市(R05.07.30)



※ ゴールは「スタート位置」と同じです。

※ コール走行コースは地図に記載してありません、現場の平コーンを確認してください。

※ 探査TXは、高校男子=①②×④⑤ / 高校女子=①②③×⑤ / 中学生=①②③④×

緊急連作先 090-&&&-&&&(大会事務局JF φ ¥¥¥)

トランジスタで UART を作ろう

トランジスタで UART を作ろう

きゅうた

トランジスタ “だけ” で UART と呼ばれる装置を作り、その方法について話そうと思ひます。僕の備忘録を兼ねているので、説明が雑な部分や、図が不足している部分もあると思いますが、雰囲気だけでも楽しんでもらえれば、と思います。紙だとリンクを打ち込まないとならず、相当不便だと思うので、リンクは最後のページにまとめて QR コードを用意しました。また、書いている最中にも、先生に質問したり、情報を集め直したり、回路の計算をしたりしているので、僕の知識量が大分変化しています。そのため、同じ話が何度も出てきたり、話がわき道にそれまくったりしますが、、、ライブ感を楽しんでください(笑)

部誌の読み方について、中学生でも一応理解できると思いますが、先述した通りそこそこ難しい話を雑に書いてるので、読んでも理解できない可能性がとても高いです。というのも、僕がどこまで説明すればいいか分からず、ある程度電子工作とか機械とかが好きでちょっと勉強したことがある感じの人に説明するつもりで書いているからですね。中学物理(・情報)を修了してから読まないと分からぬ部分がある気がします。多分同級生でも 3 割くらいしか理解してくれないですね(そもそも興味を持ってくれるのが 1 割も居ないという説もありますが、、、)。まあ、とにかく難しいわけです。必死に読み解いて「よし、僕も作ってみるぞ」ってなったらとても素晴らしいのですが、読んでるうちに挫折し「もうええわ、つまらん」ってなる可能性の方が高いと思います。なので、「なんかよう分からんけど、頑張ってんねんやな」くらいで読むのが良いと思います。もし本気で読む気があるなら、「ちなみに」から始まる文章は一旦全飛ばしでもいいと思います。話が脇道に逸れていて、一緒に読んだら訳が分からなくなることがあるからです。部誌を読みながら僕が隣で説明するとかがいいと思いますが、、、文化祭中に僕に話しかけてもらえば、連絡先か何かを交換できるかもですね、、(笑)



ダラダラと書いてきましたが「UART とトランジスタについての雑な解説」の見出しをページの頭に持て来たかっただけの文章です。紙媒体の部誌の締め切りには間に合わなかった部分がたくさんある(完成していない)ので、説明の図や実際の作品の写真、追加の説明を増やした完全版(?)を下のリンク・QR コードに載せております。そこだと、リンクに飛びやすくなっていると思うので紙媒体で読む場合でも同時に開いて読むといいかもしれません。

<https://drive.google.com/drive/folders/1VuzjE5VHBiYyXWciyEkICH9AtbFesleQ?usp=sharing> (#0) 印刷の解像度によっては読み取りが厳しいかも知れないで最後の方のページには大きい QR コードを用意しておきました。

トランジスタで UART を作ろう

目次もどき…55

- > UART とトランジスタについての雑な解説…56
 - ・UART とは…56
 - ・トランジスタとは…56
- > データシートを読もう…57～60
 - ・実際にデータシートを読んでいこう…57
 - ・データシート 1 枚目の上部…57
 - ・データシート 1 枚目の下部…58～59
 - > Absolute Maximum Ratings について…58
 - > Electrical Characteristics について…58～59
 - ・データシート 2 枚目のグラフ群…60
 - > 上段右側のグラフについて…60
 - > 中段右側のグラフについて…60
 - > 残りのグラフについて…60
 - ・データシートの残りのページ…60
- > 実際に設計しよう…61
 - ・論理回路とは…61
 - ・実際に論理回路で DFF を設計する…62～65
 - > DFF とは…62
 - > D ラッチ回路を設計しよう…62～64
 - > DFF を設計しよう…64～65
 - ・実際にトランジスタで NOT 回路・NOR 回路・NAND 回路を設計する…66～73
 - > 回路上での記号について…66
 - > トランジスタで NOT 回路を設計しよう…67～70
 - > トランジスタで NOR 回路・NAND 回路を設計しよう
 - > 余談「なぜ OR 回路や AND 回路を直接作らないのか」…73
- > 終わりに…74
- > QR コード一覧…75

トランジスタで UART を作ろう

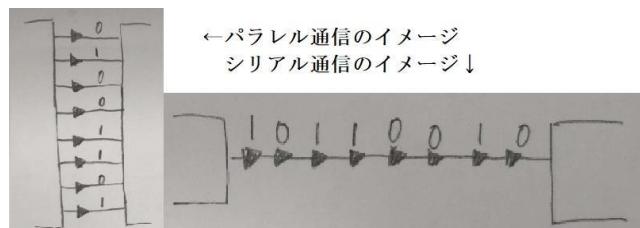
UART とトランジスタについての雑な解説

> UART とは

皆さん、USB は使ったことがあると思います。USB ではデータの通信をすることが出来ると 思いますが、その元になったものとでも考えてください。

少し詳しい話をすると UART も USB もシリアル通信という通信方式で、USB は通信方式としてはかなり複雑なので、簡単なコンピュータでは UART を使って、USB に変換する装置で USB に変換し、通信を行っています。シリアル通信について解説もしておきます。例えば

01001101(2進数で 77)を通信する時、一つの方法として考えられるのは、通信ケーブルを 8 本用意して、各々が 0 か 1 を送るという方法が考えられると思います。ただ、これだと、桁数が 64 桁や 32 桁になると本数が増えて鬱陶しかったり、ノイズが乗りやすくなったりします。そこで 0、1、0、0、、、と順々に送っていく方式が考えられました。前者をパラレル通信、後者(UART や USB)をシリアル通信と言います。

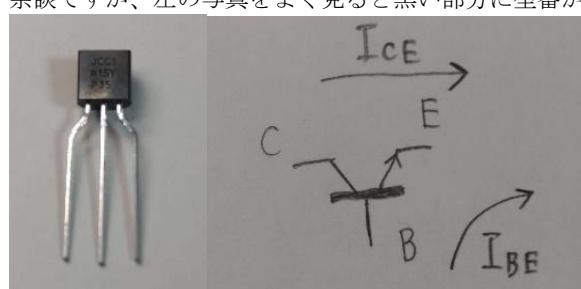


> トランジスタとは

今回使っていくのは KSC1815Y という型番(名前)の NPN 型トランジスタです。トランジスタには三つの端子があって各々にベース、エミッタ、コレクタという名前が付いており、NPN 型のトランジスタは「ベースからエミッタに少しの電流を流すと、コレクタからエミッタにたくさんの電流が流れる」という仕組みになっています。

左の写真が実際のトランジスタの外観で、回路図では右の写真のように描かれます。B と書かれているのがベース、C と書かれているのがコレクタ、E と書かれているのがエミッタで、B→E と流れている電流を I_{BE}、C→E と流れている電流を I_{CE} と呼び、I_{CE} の値が I_{BE} の値の一定数倍になります。その定数がトランジスタによって異なっているという感じです。

余談ですが、左の写真をよく見ると黒い部分に型番が書いてあります(見えない)。



トランジスタで UART を作ろう

データシートを読もう

トランジスタ回路を設計していく上で、トランジスタの詳細な情報を知らなければ回路を設計できません。そのため、設計する前の段階として、データシートと呼ばれるトランジスタの仕様書・説明書を読んで、使うトランジスタについての情報を集めなければなりません。僕もそこまで詳しく理解していないので、今回改めて調べ、ここにまとめておきたいと思います。

> 実際にデータシートを読んでいこう

トランジスタの型番を検索すると色々なデータシートを見ることができ、それを元にトランジスタ回路の設計をしていくことになります。今回使うトランジスタは KSC1815Y なので Google かなんかで「KSC1815Y」と検索してみてください。すると、以下のようなサイトがヒットすると思います。実際に読んでいきましょう。

[https://pdf1.alldatasheet.jp/datasheet-pdf/view/53347/FAIRCHILD/KSC1815Y.html \(#1\)](https://pdf1.alldatasheet.jp/datasheet-pdf/view/53347/FAIRCHILD/KSC1815Y.html)

[http://doc.chipfind.ru/html/fairchild/ksc1815y.html \(#2\)](http://doc.chipfind.ru/html/fairchild/ksc1815y.html)

データシートは大体英語で書いてありますが、読めばいいところだけ済うような読み方なら、数学の x や y といった記号がローマ字であるのとあまり変わらないので「うわ、英語かよ」ってなるほどではないです。英語にアレルギーがある人は TOSHIBA が公開しているデータシート等もあるので是非↓(読み方について解説しているサイトも載せときます)
[http://www.op316.com/tubes/tips/image/2sc1815.pdf \(#3\)](http://www.op316.com/tubes/tips/image/2sc1815.pdf)

[https://start-electronics.com/electronics/elec/read-datasheet/ \(#4\)](https://start-electronics.com/electronics/elec/read-datasheet/)

> データシート 1 枚目の上部

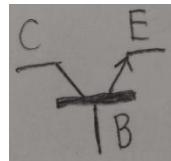
まず、「Complement to KSA1015」と書いてありますが、これは「KSA1815」の互換品であることを示しています。型番の意味をざっくりと解説すると日本の型番の場合、「2」がトランジスタ(1 だとダイオード)という事を表し、「2SC」は NPN 型(2SA だと PNP 型)、1815 は半導体素子の登録番号(出席番号的なもので特に意味を持たない)、そして最後の「Y」は後述する増幅率のランクを表しています(O : 70-140, Y : 120-240, GR : 200-400, BL : 300-700)。他にもアメリカの型番のルールがあつたり、半導体メーカーが独自に命名した型名もあつたりします。ちなみに今回実際に使ったトランジスタは「JCC1815Y」と書いてあり、調べた感じ onsemi という企業が付けた KSC1815Y の独自の型名っぽいです(KSA1015 の互換品 KSC1815Y の互換品ってことですかね、、、)。

また、データシート上部右側の絵は、トランジスタの平たい面を上に向けて、端子が左からエミッタ、コレクタ、ベースであることを示しています。回路上の記号でのエミッタ、コレクタ、ベースの並び方は先述した写真で固定ですが、部品での並び方はトランジスタによって変わり、これを正しく理解し設計しないと全く動かないで、データシートで一番大事と言っても過言ではないかも知れません。

トランジスタで UART を作ろう

- > データシート 1枚目の下部
- > Absolute Maximum Ratings について

「Absolute Maximum Ratings (絶対最大定格)」とは「これ以上流すとトランジスタがぶっ壊れるよ」の値を表しています。



V_{CBO} はベースコレクタ間にかけることのできる最大電圧、 V_{CEO} はエミッタコレクタ間にかけることのできる最大電圧、 V_{EBO} はベースエミッタ間にかけることのできる最大電圧を表しています。以下のサイトで詳しく説明してあります。

[https://detail-infomation.com/bipolar-transistor-absolute-maximum-rating-rated-voltage/ \(#5\)](https://detail-infomation.com/bipolar-transistor-absolute-maximum-rating-rated-voltage/)

I_C はコレクタに流せる(C→E)最大電流、 I_B はベースに流せる(B→E)最大電流、 P_c はコレクタで消費される最大電力($C \rightarrow E$ の電流×CE 間電圧)を表しています。

$T_J \cdot T_{STG}$ はそこまで気にしなくていいと思います(使う時に温度が上がる所以、その最大温度と保管時の温度について書いてあります)。 P_c を守っておけば勝手に守られるものと考えてもいいです(そして P_c もそう超えることはないので、、、)。

ここでまず重要なのは、 $I_C \cdot I_B$ 、そして一応 V_{CBO} 、 V_{CEO} 、 V_{EBO} が重要だと思います。一応というのは乾電池で回路を作るなら、乾電池(2個分)の 3V を越えることは無く、マイコン等から電圧をかける場合も大体 3V~5V であり、今回の場合 V_{CBO} 、 V_{CEO} 、 V_{EBO} はすべて 5V 以上であるからです。

- > Electrical Characteristics について

I_{CBO} とは「Collector Cut-off Current (コレクト遮断電流)」と書いており、これはエミッタをオープン(回路として繋がっていない状態)にした時に本来流るべき方向ではない方向に流れてしまう電流の値です。 I_{EBO} とは「Emitter Cut-off Current (エミッタ遮断電流)」と書いており、これはコレクタをオープンにして EB 間に逆電圧をかけた時に漏れてしまう電流の値です。今回の回路の場合、エミッタは接地(アースに繋ぐ(電池で言うと - (マイナス)側に繋ぐ))であり、逆電圧はかかることが無いので I_{CBO} 、 I_{EBO} ともにあまり重要でないと思います。

[https://detail-infomation.com/bipolar-transistor-collector-cut-off-current/ \(#6\)](https://detail-infomation.com/bipolar-transistor-collector-cut-off-current/)

[https://detail-infomation.com/bipolar-transistor-emitter-cutoff-current/ \(#7\)](https://detail-infomation.com/bipolar-transistor-emitter-cutoff-current/)

hFE 、これがトランジスタの肝となります。これはエミッタ接地の場合の電流増幅率のことを表しています。つまり ' $hFE = I_C / I_B$ ' ということです。表に書かれている「Test Condition (測定条件)」以外を知りたい場合は、データシート 2枚目の真ん中の段の左側のグラフを見ます。先に出したデータシート(#1・#2)は見づら過ぎるので下のデータシート(#8)の 3枚目の中段左側のグラフを見た方がいいかもしれません。これは CE 間電圧が 6V として、コレクタ電流を変えた時に hFE がどうなるかを示しています。

トランジスタで UART を作ろう

今回の回路では CE 間電圧を 3V にする予定ですが、このグラフにはありません。ただ、CE 間電圧が小さい場合は hFE が落ち始めるのが早くなるだけなので小さい電流だけ流すようにすればこの落ちる現象は無視して hFE は一定値として使うことが出来ます。詳しくは#9・#10・#11を見てください

<https://www.onsemi.com/pdf/datasheet/ksc1815-d.pdf> (#8)

<https://detail-infomation.com/bipolar-transistor-hfe-ic-characteristics/> (#9)

<http://www.op316.com/tubes/tips/semicon12.htm> (#10)

<https://electronic-circuit.com/transistor-hfe/> (#11)

で、結局 hFE はどれくらいの値になるのかというと、下の「hFE Classification」に書いてあります。トランジスタの型番の末尾に付いているローマ字(?)を見ると、今回は「KSC1815Y」つまり「Y」なので、hFE は 120~240 となります。このばらつきは個体差によるものです。120~240 と幅があつて「どうやって設計するんだ」と思うかもしれませんのが、最小値・最大値のどちらでも回路が動くように頑張って設計します。

V_{CE} とは CE 間の順方向電圧です。CE 間に最低でもこの V_{CE} の電圧がかかっていないと CE 間には電流が流れず、また、電流が流れた場合、V_{CE} 分の電圧降下が発生します。

V_{BE} とは BE 間の順方向電圧です。先述した CE 間の話と同じです。ダイオード(・LED)にもこの順方向電圧があるので、調べる際は「LED 順方向電圧」と調べるとよいと思います。説明がかなり雑になってしましましたが、この V_{CE}・V_{BE} はデータシートで得られる情報で最も重要なと言つても全く過言ではありません。次の章で説明します。

f_T とはトランジション周波数であり、hFE が 1 となる周波数、トランジスタが増幅できる最大の周波数です。今回は直流でトランジスタを使うので関係ないです。#12 が参考になります。

https://www.jr3tgs-homebrew.net/contents/ft_mesurement_tool/ft_mesurement_tool.html (#12)

Cob とはトランジスタの構造上、CE 間に電圧をかけるとコンデンサのような状態になり、その時の静電容量の値となります。遅延等に関与していますが、今回は気にしなくていいと思います。#13 が参考になると思います。

<http://okawa-denshi.jp/techdoc/3-3-8TrCob-tr.htm> (#13)

NF とは「noise figure (雑音指数)」の略称です。トランジスタ内で熱によるノイズ等が発生し、トランジスタはそれも増幅してしまいます。それらの比を取ったものが NF ですが、今回はトランジスタを増幅装置としては使わないので関係ないです。#14 が参考になると思います。

<https://www.stack-elec.co.jp/?p=255> (#14)

結局ここで重要なことは、hFE が Y なので 120~240 であることと V_{CE} の最大値が 0.25V、V_{BE} の最大値が 1.0V であることです。

トランジスタで UART を作ろう

› データシート 2 枚目のグラフ群

› 上段右側のグラフについて

トランジスタはダイオードを組み合わせたような作りになっており、ダイオードと同じように BE 間と CE 間に順方向電圧がかかります。BE 間に電流が流れていらない場合は CE 間には電流が流れないこともトランジスタの特性です(今更)。順方向電圧とはざっくり言うと、「順方向電圧以上の電圧をかけると電圧が流れ始めるよ」というものです。ダイオードの場合 VF と表され、整流用のダイオード等だと 0.6V、発光ダイオード(LED)だと赤色や黄色で約 2V、白色や青色で約 3.5V です。ダイオードに順方向電圧以上の電圧をかけた場合、順方向電圧分の電圧降下を伴って電流が流れます(完全にただの物理なので各自調べて貰いたいです)。

と、余談がすぎましたが、上段右側のグラフはまさにそれを表しています。BE 間に 0.6V~0.8V をかけると CE 間に急激に電流が流れ始めるという事です。つまり 0.6V ~0.8V が BE 間の順方向電圧ということですね。NPN 型トランジスタとダイオードの素子の並び方は右上の写真のようになっているので当たり前っちゃ当たり前です。

› 中段右側のグラフについて

先述した通り、hFE はコレクタ電流が大きくなると下がってしまうことを示しています。また、CE 間電圧が小さい場合は hFE がコレクタ電流を上げていったときに早めに落ち始めますが、そもそもコレクタ電流には 10mA も流さないように設計するので下がる現象は無視でいいと思います。

› 中段右側について

ここは今回の回路設計において 1 番重要な要素です。抵抗値を求める計算に使います。特に VBE の最小値・最大値、そして VCE の最大値はマストです。今回、コレクタ電流は 10mA にする予定なので、グラフ s によると VCE の最大値は 100mV となるはずですが、余裕をもって表の最大値 0.25V を採用してもいいと思います(僕は表の 0.25V を採用しました)。VBE は最小値が 0.6V(これは上段右側のグラフと同じであると言えるでしょう)、最大値が微妙ですが、表の 1.0V を採用しましょう。余談ですが、グラフの縦軸は間隔が一定でありません。これは対数グラフだからです(よく見てください、原点も(0, 0)じゃないです)。読み取りは通常通りなので安心して読んでもらって構いません。

› 残りのグラフについて

特に気にしなくていいと思います。

› データシートの残りのページ

大きさや、製品を使うにあたっての注意事項、商標・責任などが書いてありました(多分)。個人で使う分には気にしなくていいはずです。

実際に設計しよう

ここでは今までデータシートで集めたデータを元に実際にトランジスタ回路を設計していきましょう(言っても使うのは hFE と $V_{CE} \cdot V_{BE}$ だけなんですが、、、)。

設計していく回路は論理回路、DFF、UARTといった順にだんだんとレベルアップしていきます。全部 0 から説明することも出来るのですが、僕が特に難しいと思う「トランジスタによる論理回路の設計」「UART の考え方」について説明したいと思います。ちなみに難しいと思う理由は、「ネット上であまり情報を見かけない」かつ、見つけたところである程度知識がある状態の人をターゲットにしているのか、「説明が不足している・理解できないことが多い」からです。

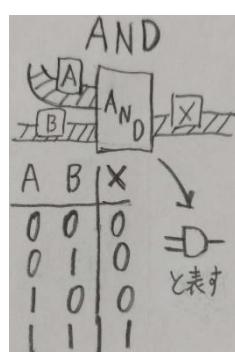
> 論理回路とは

最近は情報の授業で扱われている、論理回路。ネット上での解説もたくさんあると思いますが、一応雑に説明したいと思います。ちなみにトランジスタ回路で直接作ることが出来る回路は NOT 回路、NAND 回路、NOR 回路です。ここで「直接作ることが出来る」というのはトランジスタの個数が 1, 2 個で作ることが出来るということです(お馴染みの OR 回路や AND 回路は 3 個必要になる)。もちろんそれらを組み合わせることでどんな回路でも作ることが出来ます。

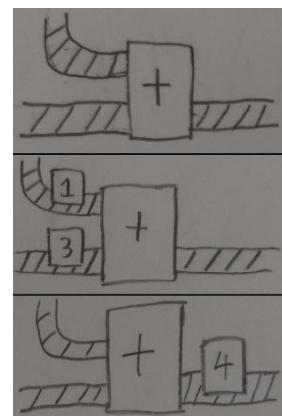
論理回路というとなんだかすごい大層な装置であるように感じてしまいますが、ただのトンネルみたいなものと考えていいです。入ってきた値を加工して送り出す、そんなトンネルのようなものです。

例えば、足し算というトンネルを考えてみましょう。右図のように、左側の 2 つのベルトコンベアから値を受け取って右側の 1 つのベルトコンベアから受け取った 2 つの値を足し合わせた値を出すようなトンネルです。例えば、1 と 3 という値が入ってきたら、4 を出すみたいなトンネルになっています。

論理回路も受け取った値の加工方法が違うだけで、やっていることはさっきの説明と何も変わりありません。「AND 回路は入ってきた 2 つの値がどちらも 1 の時、1 を出し、それ以外の場合は 0



を出す。OR 回路は入ってきた値がどちらか 1 の時、1 を出し、それ以外の場合は 0 を出す。NOT 回路は入ってきた値が 0 なら、1 を出し、それ以外の場合は 0 を出す。」というだけです。それをまとめた表を左に示しておきます(AND だけですが、、、)。



トランジスタで UART を作ろう

> 実際に論理回路で DFF を設計する

論理回路を実際に使って設計していきましょう。学校の授業だと、AND 回路、OR 回路、NOT 回路等を使って XOR 回路を作ると思いますが、その応用版みたいなものです。

> DFF とは

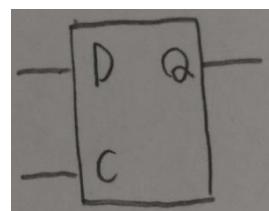
DFF、、、D フリップフロップ回路とは、値を保持しておく回路です。今までの NAND 回路や NOR 回路、NOT 回路は、ある値の入力を受け取っている間、その値によって計算された値を出力しているにすぎず、入力を変えてしまえば出力も変わり、前回の値を留めておくというようなことは出来ません。

なぜ、前回の値を留めておく必要があるのかというと、それが無ければ人間による値の変更でしか値が変わらなくなるからです。コンピュータというものは論理回路の集合でしかなく、論理回路は入力によって値が一意に定まる(ランダムには変わらない)ので、人間が操作した結果(例えば、写真を開くとかメールを開くとか)をコンピュータが実行する場合(例えば、写真をディスプレイに表示するとかメールの内容を表示するとか)、各々の操作内容と実行結果を対応させる超絶巨大な論理回路を組み上げる必要があると思います(その場合、最早コンピュータではなく、一つの巨大な回路と言った方がいいかもしれませんね)。しかし、前回の値を留めておくような装置があれば、何回も同じ動作を繰り返すような操作が出来たり、前回の状態によって次の操作を変えるような操作(プログラミングで言う if 文的なもの)が出来たりするようになって、コンピュータを作成することが出来ます。ここで僕が語ったことは間違いを含んでいるかもしれない、雑に流していただきたいです。とにかく、DFF とは値を保持しておく回路だということです。

> D ラッチ回路を設計しよう

DFF を作る前に D ラッチ回路を設計しましょう。DFF とは D ラッチ回路を 2 つ組み合わせて作られるものであり、D ラッチ回路を設計することは、この章(「実際に論理回路で DFF を設計する」)で最も重要な話だと思います。ちなみに、今回僕が設計した D ラッチ回路はもしかしたら D ラッチの定義とは少し違うかもしれません(が、本質は同じだと思うので、今回は僕が決めた定義で行きます)。

D ラッチ回路は、入力を 2 つ、出力を 1 つ持つ回路です。入力 2 つの内、1 つは書き込み状態と保持状態を確認する値、もう 1 つは書き込む値、そして出力は保持している値です(本来の D ラッチ回路だと、保持している値を反転した値も出力にあります)。また、回路の記号としては右のようなものを使いたいと思います(入力する値は D, C、出力は Q とします)。



トランジスタで UART を作ろう

入力 C が 1 の時に書き込み、0 の時に保持として設計していきます。つまり、C に入れる値を 1 にした時に、Q の値を D にして、C に入れる値を 0 にした時には D の値が何であろうと Q の値は変更しないようにするということです。例えると、箱と蓋みたいなことでしょうか。「C が 0 の時は、蓋を閉めて箱の中身を変えないようにする。C が 1 の時は、蓋を開けて、箱の中に D を入れる。箱の中身は Q として表示する。」みたいな感じです。AND 回路のように、対応表を書いておきたいと思います(右図)。

では、実際に設計していきたいと思います。

1. C が 0 の時、D が何であっても Q(出力)に何も影響しないようにしたいので、AND 回路を使います。AND 回路に C と D を入力で渡すと、C が 0 の時、D が 0 だろうと 1 だろうと、出力は 0 になります。ちなみに、本来は出来るだけトランジスタの数を減らすために NAND 回路で実装できるか(DFF を作れるか)を試すべきですが、今回の場合ここに NAND 回路を使っても、他の部分で使う NOR 回路(※1、後述)が OR 回路じやないといけなくなり、結局全体のトランジスタ使用数は変わらないので AND 回路を使うことで話を進めていきます。

2. C が 0 の時、Q の値(前回の Q)を再び Q(出力)に入れなければなりません(保持は再代入によって実装したいと思います)。逆に、C が 1 の時は Q(出力)に何も影響しないようにしたいので、NOR 回路を使います。OR 回路でも本質は変わらないのですが(NOR 回路は入力の片方が 1 の時、絶対に 0 を返し、OR 回路は絶対に 1 を返すというのが味噌です)、トランジスタの節約のために NOR 回路を使います。ちなみに、先述した※1 の NOR 回路というのがこの NOR 回路です。

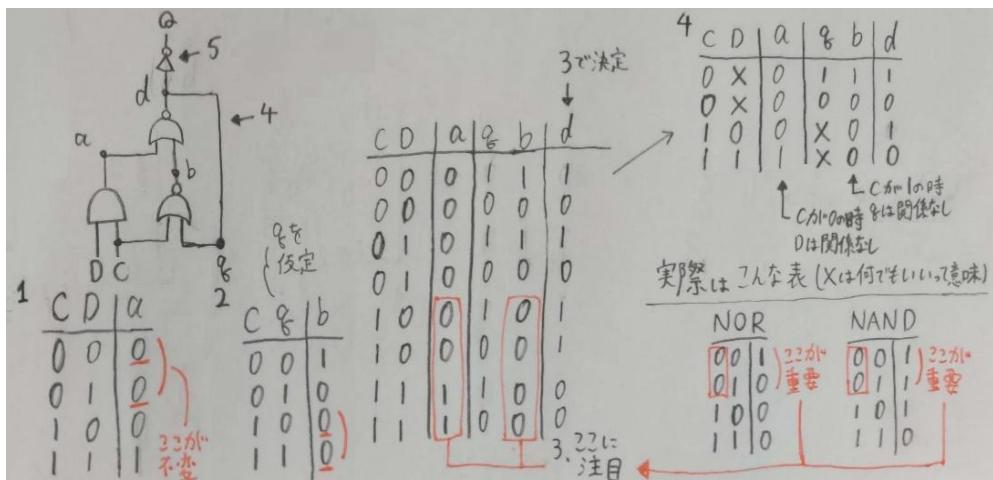
3. 1 と 2 によって、a と b が決まりますが(下図)、C が 1 の時、b が 0 になっています。C が 1 の時は書き込みなので D によって変動する a を(d に)継承する必要があります。なので、b が 0 の時、a によって d が変化する、NOR 回路を使います(トランジスタ節約のため、NAND 回路か NOR 回路かで考えています)。

4. q の値は、d の値をそのまま入れればいいことになります。表において C が 1 の時の q の値が変わってしまいますが、それによって b は変わらないので問題ありません。

5. C が 1 の時の値に注目すると、d を反転した値が Q になります。

D-LATCH			
C	D	前回の Q	Q
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

トランジスタで UART を作ろう



このような感じで論理回路を設計していきます。重要なことは、論理回路の対応表において、片方の入力によって出力が固定されている状態に注目することです。他にも T フリップフロップ回路という、1 が入力されるたびに出力が 0 と 1 を反転する回路があるので、それを設計してみると面白いかもしれません(後で使います)。#15 はラッチ回路に関する Wikipedia、#16 は僕が参考にした T フリップフロップ回路に関するサイトです。参考になると思います。

<https://ja.wikipedia.org/wiki/%E3%83%A9%E3%83%83%E3%83%81%E5%9B%9E%E8%B7%AF> (#15)

<https://www.krrk0.com/t-flip-flop/> (#16)

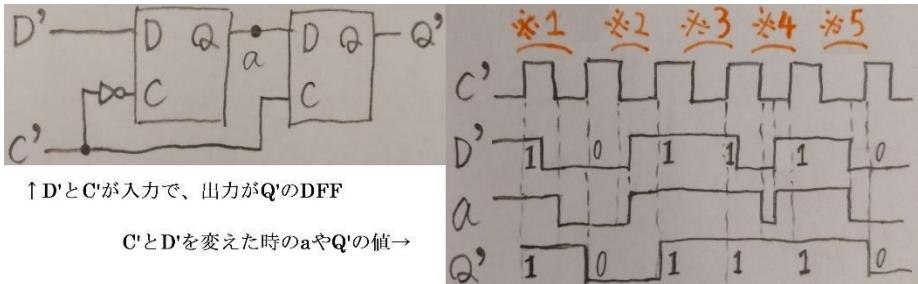
ちなみに D, C, Q という端子の名前は DFF を回路図で描く時によく使われる所以僕もそれを使いました。D はデータ(DATE)、C はクロック(CLOCK)、Q は、、なんでしょうね。クロックというのはコンピュータ全体で共有されている時計みたいなものです。

> DFF を設計しよう

さて、ようやく DFF、D フリップフロップ回路を設計していくことになります。D ラッチ回路だと C が 1 の時はずっと Q の値と D の値が同じになります。これだと UART、というかシフトレジスタでは使いづらいので DFF にします。DFF とは、C が 0 から 1 になる時のみ Q の値を D にします。ちなみに 0 から 1 になる時を立ち上がりエッジと言い、0 から 1 になることを立ち上がると言います。また、1 から 0 になる時を立ち下がりエッジと言い、1 から 0 になることを立ち下がると言います(個人的にはだいぶ変な言葉だなと思います)。どのように DFF を作ろうかとなったときに、D ラッチ回路の C に RC 遅延回路と XOR 回路を付けるみたいな対策方法も考えられますが、これだと RC 回路の遅延に若干依存すると思われる所以やめましょう(早口)。

トランジスタで UART を作ろう

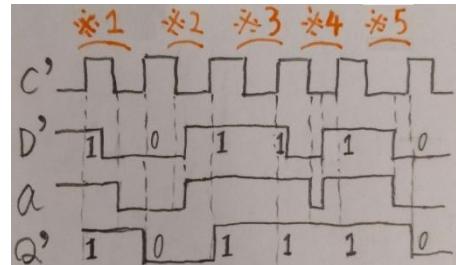
ではどうするのかというと、C に NOT を繋げた D ラッチ回路の Q を 2 つ目の D ラッチ回路の D に繋いで DFF を作ります。もう完成です。回路図になると下図左側のようになります。また、タイミング図を見ると、立ち上がりエッジの際に値が書き込まれ、



それ以外の時は保持されていることがよくわかると思います。

タイミング図について雑に説明すると、横軸に時間がとられており、線が上に上がっている時が「1 が入力・出力されている状態」、線が下に下がっている時が「0 が入力・出力されている状態」となります。なぜこのように上下で表すのかというと、0 と 1 をコンピュータでは電圧の LOW・HIGH で表しているからです。

タイミング図をよく見てみましょう。米印が付いた場所について順々に説明していくと思います。図が前ページになったので、タイミング図だけをこちらにも載せておきます(右図)。



※1について、C'が 1 の間に D'が変わったとしても a や Q'は変わらず、C'が 0 に落ちた瞬間 a の値が D'の値になります。また、C'が 0 に落ちたとしても Q'の値は変わりませんが、C'が 1 に上がった時に Q'の値が a の値になります。

※2について、C'が 0 の間に D'が変わった場合は a の値はすぐに D'の値になりますが、Q'の値は変わりません。次に C'が 1 に上がった時に Q'の値が a の値になります。※5 でも同じようなことが言えます。※3について、C'が変わっても D'が変わらなければ、a の値も Q'の値も変わりません。

※4について、※1で見たように、C'が 1 の時は D'の変化は a の値に影響せず、また※2で見たように、C'が 0 の時は D'の変化は即座に a の値に影響します。C'が 1 に立ち上がる瞬間の D'の値が Q'の値になることが分かります。

また、全体を見た時に、C'が 1 に上がった瞬間の D'の値と Q'の値に注目すると一致していることが分かると思います。まとめると、C'が 1 に立ち上がった瞬間に入力していた値が次に C'が 1 に立ち上がるまで固定される、、、という回路になります。

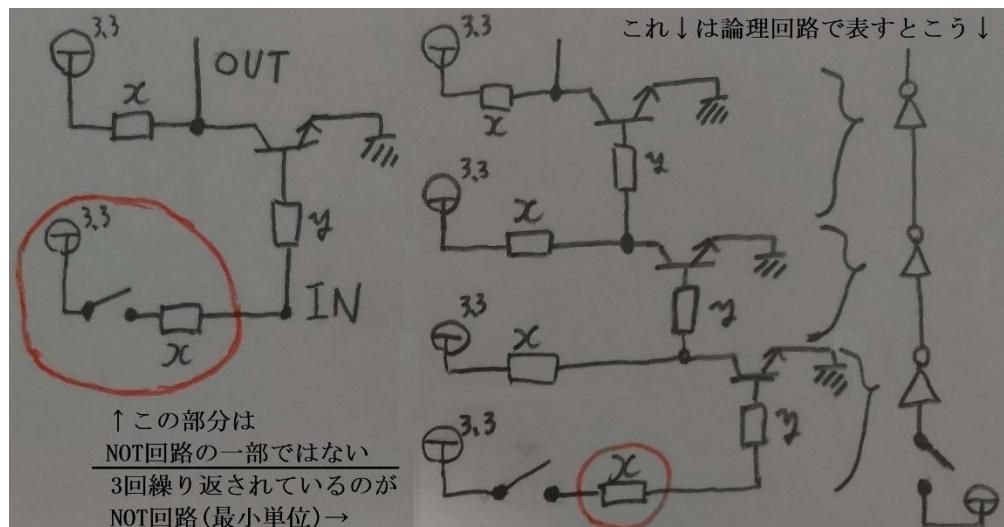
トランジスタで UART を作ろう

> 実際にトランジスタで NOT 回路・NOR 回路・NAND 回路を設計する

それでは実際にトランジスタで論理回路を設計していきましょう。データシートによつて得た hFE ・ V_{CE} ・ V_{BE} を使って、抵抗値を決めていく作業です。回路でのトランジスタの使い方を難に説明すると、(BE 間に電流を流すことで)トランジスタが発動して流れる CE 間の電流による電圧降下を利用し、NOT 回路を作り、NOR 回路・NAND 回路はその応用となっている、、、という感じです。また、なるべく同じ値の抵抗を使いたいので、NOT 回路・NOR 回路・NAND 回路全て並行して計算し、抵抗値を決定させます。

> 回路上での記号について

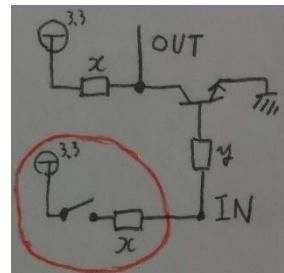
まず回路の記号について少し説明した後で実際の回路の意味について説明したいと思います。回路を見てみましょう(下図)。回路上で左側の方に描いてある「丸の中に T のようなものが描いてある記号」は電池の + と考えてもらって構いません。今回は電源をマイコンから 3.3V で供給するので 3.3 と書いてあります。次に x や y と書いてある隣に描いてある「四角い箱のような記号」は抵抗です。最後に「横棒の下にジャジャっと線が引いてある記号、、、照れてる感じになっている記号」は電池の - と考えてください。他に、トランジスタとスイッチがありますが、トランジスタは始めの方に説明したのと、スイッチは分かることと思うので説明を割愛します。なお、今回の場合、「丸の中に T のようなものが描いてある記号」はすべて同じ電池の + に接続されており、「照れてる感じになっている記号」はすべて同じ電池の - に接続されています。



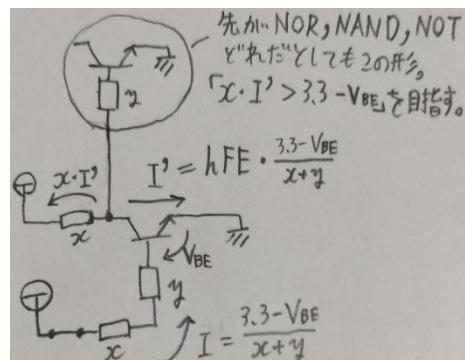
トランジスタで UART を作ろう

> トランジスタで NOT 回路を設計しよう

では、回路の意味について説明していきます。回路上でスイッチになっている部分は実際には NOT 回路や NOR 回路、NAND 回路によって制御されているのをわかりやすくするために描いているだけです。実際には論理回路(トランジスタ)が付いているというのは、OUT と書いてある部分が IN に繋がっている、、、みたいな感じです。最小単位を抜き出して右に描いてあるってわけですね(丸で囲まれている部分は本来 NOT 回路には関係ない部分ですが、計算するにあたっては必要なので書いています、、、丸に囲まれている部分と、丸の上の部分がとても似ていることに注目してください)。実際にこの回路が使われている状態を見ると言いたいことが分かるかもしれません(前ページの図の右側)、抵抗 x と抵抗 y、そしてトランジスタによって形成されているのが NOT 回路なんですね。ちなみに前ページの右側の図において丸で囲まれている抵抗 x は入力の値(電圧)が他の部分と同じになるようにしているだけで本質的には関係ないです。図が前ページになったので、重要な部分だけをこちらにも載せておきます(右図)。



この先、ほぼ文章ごり押しの説明が続き、分かりづらいと思ったので改行を多めに取りましたが、それでも初見だと意味不明だと思います(説明がかなり不足しているので...)。なので、先に求められた式を見て(説明の後に書いてあります)、その意味を下の文章を読んで読み取る...のような方法が良いと思います。右図も参考にしてください。



実際にスイッチを ON にした場合を考えましょう(NOT 回路に 1 を入力した場合と状況は同じです)。この場合、 $(3.3-V_{BE})/(x+y)$ A(アンペア)の電流が B から E に流れます(抵抗 x の抵抗値を x、抵抗 y の抵抗値を y にしています)。

その電流をトランジスタが增幅し、C から E に $hFE \times (3.3-V_{BE})/(x+y)$ A の電流が流れようとします。NOT 回路なので、スイッチが ON になった場合は OUT には電流が流れていません。OUT の先には前ページの図の右側・上図(下の方)で確認できるように、トランジスタがついています。

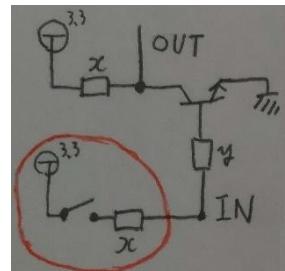
電流が流れないようにするために右図の上側の抵抗 x による電圧降下で OUT の先に付いている V_{BE} 以下の電圧にしてしまえばいいです。

トランジスタで UART を作ろう

また、 x による電圧降下でトランジスタの V_{CE} 以下の電圧になってしまったら、トランジスタに電流が流れない状態になるのかとも思うかもしれません、恐らくその場合は hFE を守らず、 $(3.3-V_{CE})/x$ A の電流が C から E に流れると思います。するとそれは本来流れるはずだった電流 $(hFE \times (3.3-V_{BE})/(x+y))$ A よりも小さく、 x による電圧降下が低くなり OUT が、、となつて複雑になるので、 x による電圧降下で右図のトランジスタの V_{CE} 以下の電圧にならないように気を付けます。

ちなみに、OUT の先にトランジスタが付きますが、その前に抵抗 y が付きます。じゃあ、 x と y による電圧降下で V_{BE} 以下にすればいいのではないかと思うかと思いますが、その場合は、 x での電圧降下では OUT 側に電流が行くのを抑えることが出来ていないので、 $(3.3-V_{BE})/(x+y)$ A が OUT の先のトランジスタに流れます。そして $hFE \times (3.3-V_{BE})/(x+y)$ A の電流が C から E に流れます。この時結局抵抗 x に流れている電流は $(3.3-V_{BE})/(x+y) + hFE \times (3.3-V_{BE})/(x+y) = (hFE+1) \times (3.3-V_{BE})/(x+y)$ A です。この際、抵抗 x による電圧降下は $x \times (hFE+1) \times (3.3-V_{BE})/(x+y)$ V であり、少なくとも今回のトランジスタの $V_{CE} \cdot V_{BE} \cdot hFE$ では $x \times (hFE+1) \times (3.3-V_{BE})/(x+y) < 3.3-V_{CE}$ ので、回路が成り立つてしまう、つまり、OUT に電流を流す(OUT に 1 を出力する)ことが出来てしまうので NOT 回路として破綻してしまいます。安定性も考えて x による電圧降下だけで V_{BE} 以下の電圧にし、OUT 先のトランジスタが発動しないようにしましょう。

次に考えるのはスイッチを OFF にした時の NOT 回路の動きですね。スイッチが OFF の場合、そもそもトランジスタが発動しません。そのため、右図の左上の電源は抵抗 x を通りそのまま OUT に電圧をかけることになります。特に計算することはありません。なぜならば、右図の左上の電源が抵抗 x を通りそのまま OUT に電圧をかけることになるという状況は右図の左下のスイッチを入れた状況と変わらず、先ほど、スイッチを ON にした時に計算をしたからです。



結果、どういう式が得られたのかをここに示すと、

$3.3 - V_{BE}$ の最小値 << $x \times hFE \times I < 3.3 - V_{CE}$ の最大値
$(x+y) \times I < 3.3 - V_{BE}$ の最大値

上のようになります。ここで、 I というのはスイッチが ON の際、B から E に流れる電流の値としています。

上側の式で V_{BE} の最小値を使っている理由は、式変形をするとよく分かると思うので、式変形してみましょう。式を変形すると「 $3.3 - x \times hFE \times I << V_{BE}$ 」となるわけですが、この式の意味は先述した通り、抵抗 x に流れた電流による電圧降下の結果 V_{BE} 以下の電圧になるように x の値を決めたいというものです。

トランジスタで UART を作ろう

この時 V_{BE} 以下の電圧にしているのは BE 間に電流が流れていなければいけないからであり、流れ始める最小の電圧を入れるのはごく自然に感じられませんか？（最小より大きい値にしている場合、 $3.3 - x \times hFE \times I$ が最小の値より大きくなる可能性があり、この時 V_{BE} が最小の値をとるトランジスタを使用していた場合、電圧降下の量が足らず、トランジスタが発動してしまいます）

次に、上側の式で V_{CE} の最大値を使っている理由も式変形すると「 $V_{CE} < 3.3 - x \times hFE \times I$ 」となり、抵抗 x に流れた電流によって電圧降下が起こっても CE 間に電流を流せるように V_{CE} 以上の電圧をかけたいというものなので、 V_{CE} の最大値以外を取ると、 V_{CE} が大きすぎる場合に流れなくなってしまいます（流れなくなるというより電流の値が下がるだけだと思われますが、、、まあどちらにせよ期待した結果にはなってくれません）。なので余裕も持つて V_{CE} の最大値を取っておこうということですね。

そして、下側の式が V_{BE} の最大値を使っている理由ですが、式変形すると「 $V_{BE} < 3.3 - (x+y) \times I$ 」となり、つまりこれはスイッチを ON にした時にトランジスタが発動するためには必要な条件です。ここで最大値を用いていないと、 V_{BE} がある程度大きい場合 NOT 回路の核となっているトランジスタが発動しないので困りますね。

最後に身も蓋もないことを言うようですが、基本的に不等式の範囲が小さくなるように値を設定していれば何も問題ないはずです（笑）。

では、実際に求めた式にデータシートで求めた値を入れてみましょう。

$$3.3 - V_{BE} \text{ の最小値 } << x \times hFE \times I < 3.3 - V_{CE} \text{ の最大値}$$

まず、こっち↑の式に注目します。大分前のページを見ると、、、 V_{BE} の最小値は 0.6、 V_{CE} の最大値は 0.25 です。代入すると、

$$2.7 / (hFE \times I) << x < 3.05 / (hFE \times I)$$

そして、 $120 < hFE < 240$ ですから、

$$2.7 / (240 \times I) < 2.5 / (hFE \times I) < 2.7 / (120 \times I)$$

$$3.05 / (240 \times I) < 3.05 / (hFE \times I) < 3.05 / (120 \times I)$$

となるので、結局 x に関しては以下の不等式が成り立ちます。

$$(2.7 / 120) \times (I) << x < (3.05 / 240) \times (I)$$

さて、察しの良い方ならお気づきでしょう。 $2.7 / 120 > 3.05 / 240$ となっています。つまり NOT 回路を形成することのできる抵抗 x は存在しない！ということです。

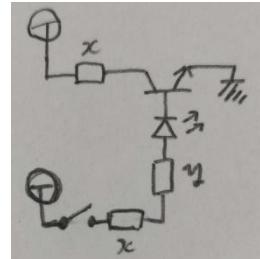
つまり、このトランジスタでは NOT 回路を作れないということです、、、まあ、電源の値(3.3V)を変える(例えば 1.2V にする)と作ることが出来るのですが、、、また、電源が 3.3V でも NOT 回路を形成できる抵抗 x を存在させることが出来るようなトランジスタを探すことも選択肢として考えられるでしょう。しかし、そのようなトランジスタを探すとなると、 $V_{CE} \cdot V_{BE} \cdot hFE$ がうまい具合になっているトランジスタを探さねばならず、トランジスタの $V_{CE} \cdot V_{BE} \cdot hFE$ が一覧になっている商品ページを見てもすぐには見つけられません(プログラム組んだら何も大変なことではなさそうですが、、、)。

トランジスタで UART を作ろう

そして、何よりもここまで書いてきた部誌のデータシートに関する部分を書き直すのが面倒であり、僕は KSC1815Y をすでにかなり購入してしまったので、我々は KSC1815Y から逃れられません。また、電源は作りやすく実験もしやすいという観点から、3.3V を採用したいです。

しかし、上の式が「電源が 3.3V かつトランジスタは KSC1815Y である状況で NOT 回路を作ることは不可能である」と証明してしまっています(hFE は 120~240 と幅があるので厳選すればそれほど問題もない、、、ま、まあ、気持ちの問題ですよ)。しかし我々は退くことも出来ません。そこで次に僕の考えた解決策を示したいと思います。ちなみに大分無茶苦茶なことをしていると思われるので、馬鹿なことしてるなあと思ってください。でも面白そうならやってみるしかないのです。

さて、解決策とは、、単に V_{BE} の値を上げちゃおうというものです。つまり $(2.7 / 120) \times (I) << x < (3.05 / 240) \times (I)$ となっているので、2.5 の値をさらに小さくすることで $(0.7 / 120) < (3.05 / 240)$ が成り立つようになります。では、実際どのように V_{BE} の値を上げるのかというと、、、LED を使います。ダイオードでも(の方が)いいのですが、なぜ LED を使ったのかを弁明すると、まずは、余ってたからですね。他にも、光によって目でデバックできる、面白そう、見た目が派手になって展示にいい、などの理由で LED を使います。



実際に回路図を書いてみると右上の図のようになります。この時、先ほどまでの式に登場する、 V_{BE} の値に LED の順方向電圧の値を足して計算すればよくなります。BE 間の順方向電圧が「 $V_{BE} + V_F$ (LED の順方向電圧)」になるわけですね。さて、今回使う LED の V_F は 2.0V ですから、先ほど求めた式に入れると、

$$\begin{aligned} 3.3 - V_{BE} \text{ の最小値} - V_F &<< x \times hFE \times I < 3.3 - V_{CE} \text{ の最大値} \\ (0.7 / 120) \times (I) &<< x < (3.05 / 240) \times (I) \end{aligned}$$

となり、この場合は $0.7 / 120 < 0.006 << 0.012 < 3.05 / 240$ なので、 x の値が存在します。同様に以下のように式を書き直して、 V_{BE} の最大値は 1.0V なので

$$\begin{aligned} (x+y) \times I &< 3.3 - V_{BE} \text{ の最大値} - V_F \\ x+y &< 0.3 / I \end{aligned}$$

となります。また、 I の値をここで決めたいと思います。データシートの 2 枚目、中段右側のグラフを見ると、コレクタ電流を 10mA 以内に収めると良さそうです。つまり、

$$hFE \times I < 10mA$$

であり、「 $120 < hFE < 240$ 」なので、 $I < 0.04mA$ ($\leftarrow I < 10mA / 240 < 10mA / hFE$)、つまり、 $I < 40\mu A$ となります。なので、今回は $I = 30\mu A$ 、 $0.030mA$ にしたいと思います。これによって、先ほどの式から、

$$\begin{aligned} 0.006 / (30 * 10^{-6}) &= 200 << x < 400 = 0.012 / (30 * 10^{-6}) \\ x+y &< 10000 = 0.3 / (30 * 10^{-6}) \end{aligned}$$

つまり、「 $200\Omega << x < 400\Omega$ 」、「 $x+y < 10k\Omega$ 」というわけです。

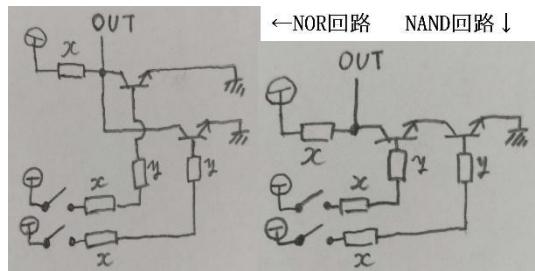
トランジスタで UART を作ろう

> トランジスタで NOR 回路・NAND 回路を設計しよう

さて、NOT 回路である程度抵抗の値を決めることが出来ました。が、全体の回路上で使う抵抗は共通化させたい、つまり NOR 回路も NAND 回路も NOT 回路で使ったのと同じ抵抗を使って製作したいので、NOR 回路・NAND 回路でも抵抗の取れる値の範囲を計算して、その条件を全て満たした抵抗を使います。ちなみに、抵抗を共通化させる目的としては、実際に回路を作っていくときに間違える可能性を下げる、大量に買った方が安い、回路上で使う抵抗を数える時に、トランジスタの個数×2, 3 で求められる、、、といった理由があります。

では、NOR 回路と、NAND 回路の回路図を見てみましょう。

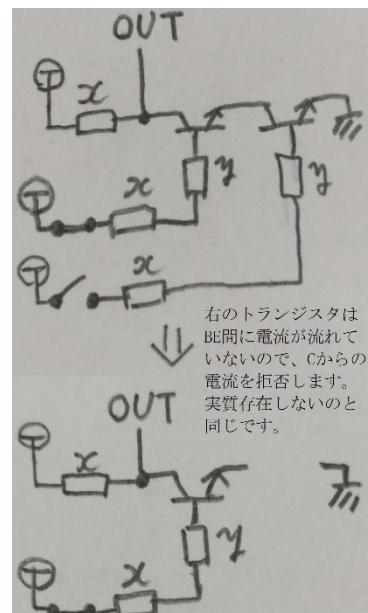
NOT 回路の OUT の部分を並列に繋いだのが NOR 回路、直列に繋いだのが NAND 回路、みたいな認識でいいと思います。



スイッチを入れてみたところを想像してみましょう。

NOR 回路の場合は、どちらかのスイッチを入れると(ON にすると)、トランジスタに電流が流れ、上図で一番上にある抵抗 x によって電圧降下が発生し、OUT が 0 になります。また、スイッチをどちらも切っていると(OFF の状態では)、トランジスタには電流が流れず、増幅されないので、抵抗 x に流れる電流はごくわずかであり、電圧降下は発生せず、OUT には高い電圧のまま出力されます(OUT は 1 になります)。

NAND 回路の場合は、どちらかのスイッチが切れていると、スイッチが切れている方のトランジスタには電流が流れず、もう片方のスイッチが入っていたとしても、そのトランジスタは回路的に切れている、電池の - 側に繋がっていないので(なぜなら、スイッチが切れている方のトランジスタが電流を流すことを拒否するからです(トランジスタは BE 間に電流が流れていなければ CE 間に電流が流れません))、トランジスタの方には電流が流れず、増幅されないので、抵抗 x に流れる電流はごくわずかであり、電圧降下は発生しないので OUT は 1 になります。逆にどちらのスイッチも入っている場合は、トランジスタが電流を増幅して流すので、抵抗 x によって電圧降下が発生し、OUT が 0 になります。(右図:「右のトランジスタには BE 間に電流が流れていないので、C からの電流を拒否します。実質存在していないのと同じです。」)



トランジスタで UART を作ろう

では、NOR回路、NAND回路で抵抗の値を計算していきましょう。

といつても、やることはほとんど変わりません。まず、NOR回路についてですが、スイッチが「どちらも切れている状態」「どちらかだけ入っている状態」はNOT回路と何も変わらないので、先ほどNOT回路で計算した結果と同じになります。なので、スイッチが「どちらも入っている(ONになっている)状態」だけを考えます。この場合、2つのトランジスタが各々 $hFE \times I$ の電流をコレクタに流そうとするので、抵抗 x には NOT 回路でスイッチを ON にした時の 2 倍の電流が流れます。つまり、

$$3.3 - V_{BE} \text{の最小値} - V_F << x \times 2 \times hFE \times I < 3.3 - V_{CE} \text{の最大値}$$

のように 2 倍されるので、得られる結果は、

$$0.003 / (30 \times 10^{-6}) = 100 << x < 200 = 0.006 / (30 \times 10^{-6})$$

となります。 $x+y$ に関する式は、NOR回路も出力が起きる場面では、「トランジスタ側に電流が流れず、電源から抵抗 x を通って OUT だけに電流が流れる」という状況は NOT回路と変わらないので、改めて考慮する必要はありません。

次に、NAND回路についてですが、これもスイッチが「どちらかだけ入っている状態」「どちらも切れている状態」は NOT回路と何も変わらず、スイッチが「どちらも入っている状態」だけを考えればいいです。この場合、トランジスタが直列に並んでおり、どう増幅されるのかと疑問に思うかもしれません、各々が $hFE \times I$ を流そうとして、単に抵抗 x にも $hFE \times I$ しか流れません。唯一変わるのが、「 V_{CE} が 2 つ分かかるようになる」ということです。つまり、

$$3.3 - V_{BE} \text{の最小値} - V_F << x \times hFE \times I < 3.3 - 2 \times (V_{CE} \text{の最大値})$$

のように 2 倍されるので、得られる結果は、

$$0.006 / (30 \times 10^{-6}) = 200 << x < 366 < 0.011 / (30 \times 10^{-6})$$

となります。NAND回路も $x+y$ に関する式は、NOT回路と変わらないので、改めて考慮する必要はありません。

さて、NOT回路、NOR回路、NAND回路と色々計算してきましたが、ようやく抵抗の値を決めることが出来ます。NOT回路では「 $200\Omega << x < 400\Omega$ 」、「 $x+y < 10k\Omega$ 」、NOR回路では「 $100\Omega << x < 200\Omega$ 」、NAND回路では「 $200\Omega << x < 366\Omega$ 」となったので、「 $x = 200\Omega$ 、 $y = 6k\Omega$ 」とします。 x が 200Ω で大丈夫なのか? と不等号をしっかりとみている人は思うでしょうが、まず、コレクタ電流を $10mA$ ほどしか流さない状況で、 V_{BE} が最大値を取ることはないと思われます(データシート2枚目中段右側より)。なので、実際に NOR回路を組んでみて、うまく動作しなかったトランジスタだけを排除していくべき問題ないと思われます。また、NOR回路の方でも 200Ω が引っ掛かっていますが、これも、 V_{CE} が最大値を取ることは今回の場合ないと思われる所以、大丈夫です。ついでに、 $x+y$ が $10k\Omega$ 以下なのになんで $6k\Omega$ にしたのかというと、余ってたからです(なんなら、抵抗として $6k\Omega$ なんてものは売っていない(少なくとも簡単には手に入らない)ので、 $6.2k\Omega$ を使っています。#17はカーボン抵抗の商品ページです)。

https://eleshop.jp/shop/c/c110311_p3/?ps=35 (#17)

トランジスタで UART を作ろう

> 余談「なぜ OR 回路や AND 回路を直接作らないのか」

こういう余談を書きまくるから、文章が読みづらいんでしょうね。。。チャットアプリでもメールでも鉤括弧を付けてたくさん情報を付け足す系の人です(こういう感じに。。。)。

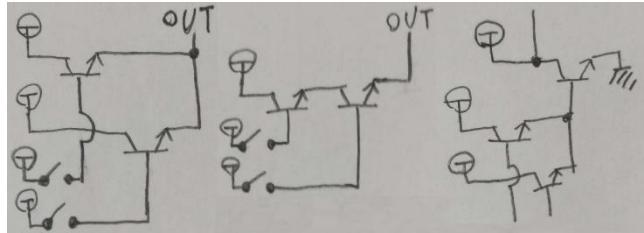
さて、ここで話したいことは「なぜ OR 回路や AND 回路を直接作らないのか」です。

トランジスタの原理を知っている・

理解した人は思うかもしれません、

「あれ? こいつ、トランジスタで NOR 回路作ってるけど、OR 回路作れんじやん」と。恐らく右図のような回路を思いついたのではないで

しょうか。上図の回路は左から、



OR 回路、AND 回路、そして一番右の図は OR 回路に NOT 回路を繋いだものです。抵抗など本質には関係ないものはだいぶ省いています(一番右はスイッチも省略しています)。実際は動いたとしても安定性が無いと思います。でも、回路を見る限りは OR 回路、AND 回路になってそうですよね。ではなぜ安定した動作が困難なのかというと。。。一番右の図に注目してみましょう。これは先述した通り、上図の一番左の OR 回路の OUT に NOT 回路を繋いだものです。例えば左のトランジスタを ON にしてみましょう。そうすると、トランジスタの CE 間に増幅された電流が流れ、NOT 回路の入力にそのまま流れます(実際は抵抗を挟むでしょうが。。。)。そして、それがまた NOT 回路において増幅されます。。。この時点で少し不味い気がしませんか? そうです、上図のような回路だとそれ単体で回路として完結していないので、正確な設計が出来ないです。回路では電池の - 側(GND)に到達するまで実際の電流や電圧がどうなるのかは計算できません。例えば、上図の一番左の OR 回路の OUT にまた一番左の OR 回路の入力の部分が繋がっていて(スイッチの代わりに OUT を繋ぐイメージです)、さらに一番左の OR 回路の入力の部分が繋がっていて。。。となつていたら、まったく GND(電池の - 側)に到達せず、どんどん増幅されていくはずです。抵抗で抑えるにしても、OR 回路だと「どちらのスイッチも ON の場合」に電流が増えますし、トランジスタの個体差がかなり響いてきます。例えば、上図の一番右側の回路では、トランジスタの増幅率は「(OR 回路のトランジスタの hFE) × (NOT 回路のトランジスタの hFE)」になるそうです(しっかりと調べられていないので間違っているかもしれません)。その場合、240×240 と 120×120 で 4 倍も差が出ます。とにかく、なるべくすぐ GND に落とす(電池の - 側につなげる)、OUT は電源から直接(トランジスタで増幅した結果を OUT に渡さない)、にしないと安定性がダメになります(設計もしやすいです)。なので、NOT 回路、NAND 回路、NOR 回路だけをトランジスタで作るようにしています。

トランジスタで UART を作ろう

> 終わりに

なんだか、中途半端な終わり方で申し訳ないのですが、ここでこの部誌は終わりになります。理由としては、「まだ実際に UART を作っていないので作ってから説明したい」「単純に提出期限に間に合わなかった」があります(後者だけでは?...)。

最初のページのリンク先に完全版を載せているので、ぜひご覧ください(このページの下にも置いておきます)。

読みづらく、分かりづらい、そして完結すらしていない。こんな文章を最後まで読んでくださりありがとうございました。

<https://drive.google.com/drive/folders/1VuzjE5VHBiYyXWciyEkICH9AtbFesleQ?usp=sharing> (#0)



この大きさなら、読み取れますかね、、、？

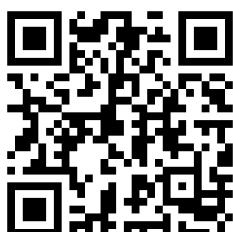
トランジスタで UART を作ろう

QR コード集

印刷の解像度によっては読み取りが厳しいかもしれません、、、その場合は完全版の方からだとリンクを押すとリンク先に行けるかと思うので、そちらを利用してください。

		
#1	#2	#3
		
#4	#5	#6
		
#7	#8	#9

トランジスタで UART を作ろう

		
#10	#11	#12
		
#13	#14	#15
		
#16	#17	#0

あとがき・その他

あとがき

ここまで読んでいただき、ありがとうございました。

アマ研の会計、K. T. です。今回部誌の編集を担当させていただいたのですが、思っていた何倍も大変でした。個人的な話で申し訳ないのですが、普段ダークモードを使っている私にとってホワイトモードで編集を一日ぶっ通してやる、というのは非常に大変で、吸血鬼が日光を怖がる理由に近しいものを感じました(笑)。編集が終わったのが 3/31 なのですが、まだ見ぬ 82 回生と新高 79 回生に思いを馳せつつ、文化祭が一日一日と近づいていることに少々恐怖を感じております。部誌の内容を見ていただいたら分かる通り、アマチュア無線研究部は様々な活動をしております。部員のやりたいことを応援するという部活の空気と仕組みがあるからだと思います。もし部誌の内容に興味を持ったこれから灘校に入学する方・在校生はぜひアマ研に入部してほしいですね(笑)。

これからも灘校文化祭をお楽しみください!進化して歓迎いたします!

Web 版記事と X(旧 Twitter) アカウントのご案内

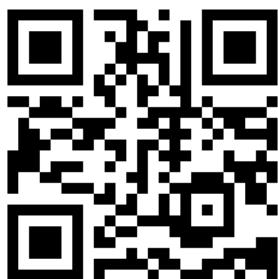
https://drive.google.com/drive/folders/1fRjjwCIUcLlro2rm3u8xZf3qhPcA_uSR

X(旧 Twitter) @JR3YYJ <https://twitter.com/JR3YYJ>

左側の QR コードから Web 版記事を閲覧・DL 出来ます。

Web 版のみで掲載している記事もあるのでぜひ。

また、右側の QR コードからアマ研公式 X(旧 Twitter) アカウントを開くことが出来ます。



発行年月日 / 2024 年 5 月 2 日

発 行 者 / 灘校アマチュア無線研究部

灘校アマチュア無線研究部 2024

いかなる場合にも第三者によるこの冊子の複製・頒布、
及びスキャンデータの送信・公開は禁止します