

TDBBU 3.1

Putting Lasing Into the Beam Breakup Simulations

K.B.Beard, D.Douglas, B.Yunn
24 August 2004

The `tdbbu 1.6h`¹ and `matbbu 2.4`² programs have been very useful predicting the beam breakup threshold in the Jlab 10kW Free Electron Laser (FEL)³⁴⁵, but neither includes the effect of the lasing process itself on the beam breakup. The latter, because it is based on finding eigenvalues of fixed matrices, does not lend itself to inserting a momentum spread on the beam, but the former, because it “follows” the bunches around the machine, does.

Very simply, `tdbbu` reads in an input file describing the system of interest as drift spaces, lenses, matrices, and cavities (in terms of their **higher order modes**, or **HOMs**) and a recirculation matrix. The X and Y axes are treated entirely independently, except for mixing that may occur in various matrices. From that input, the program generates plots showing the transverse motion of the beam as a function of time. After an initial period of settling, the motion may grow, stay constant, or dampen.

In the presence of lasing, the momentum P of the beam bunch may be changed by δP ; this changes the path length the bunch will experience and hence the time of arrival back at the HOM. For the purpose of this model, the bunch is a single point whose fractional momentum is changed by $\delta P/P$.

The time to run should exceed about 10 times the characteristic time of

¹JLAB-TN-02-045, *tdbbu 1.6: Another Tool for Estimating Beam Breakup due to Higher Order Modes*, K.B.Beard, L.Merminga, B.Yunn

² JLAB-TN-02-044, *matbbu 2.4: A Tool for Estimating Beam Breakup due to Higher Order Modes* K.B.Beard, L.Merminga, B.Yunn

³ JLAB-TN-02-042, *Estimates of the Beam Breakup Thresholds in the 10KW FEL due to HOMs*, K.B.Beard, L.Merminga, B.Yunn

⁴**JLAB-TN-03-035**, *HOM calculations for CEBAF module SL21 and NL11 in the 10kW FEL*, K.B.Beard and B.Yunn

⁵ JLAB-TN-03-034, *HOM calculations for a SuperCell Cryomodule in the 10KW FEL*, K.B.Beard and B.Yunn

the HOMs, which is about $t \sim 2^*w/Q$, so for $Q=10^5 \rightarrow T=150\mu\text{S}$, and for $Q=10^7 \rightarrow T=15\text{mS}$.

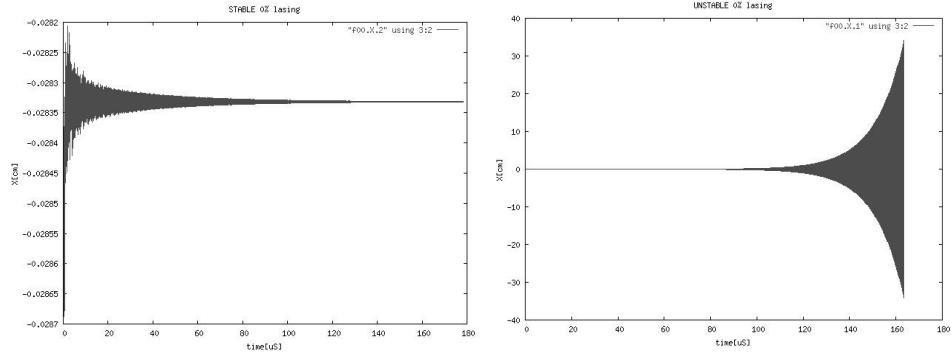


Figure 1a. An example of the transverse beam position at a point in the machine for a stable and unstable beam in the absence of lasing.

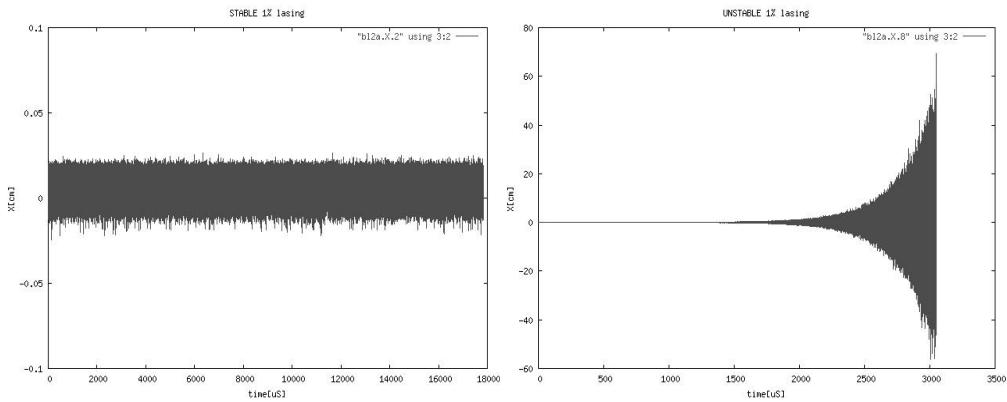


Figure 1b. The transverse position of a stable and unstable beam at a given point as a function of time in the presence of 1% dP/P lasing.

The current file for the 10kW FEL has Q_{HOMS} up to $1.19\text{E}7$, typical runs of about 17.8 mS, and each X and Y output file for each current setting has around 2.7 million lines.

`tdbbu 3.0` uses a fundamentally different algorithm than its predecessor. There are numerous changes between this version and the previous one:

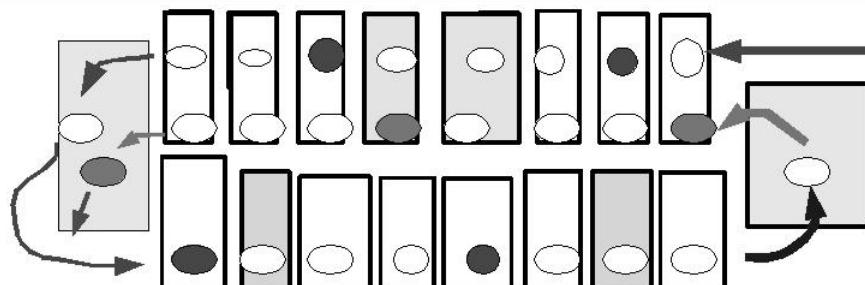
- **lasing** - a momentum spread may be introduced to simulate lasing

- **M56,T566** - path length & time dependence on $\delta P/P$
 - **beam dump** - beam dump can now be located anywhere, not just at the beginning of a recirculation matrix
 - **RF phase** - include the effect of off-crest acceleration and off-trough deceleration
 - **losses** - allows tolerable beam losses; versions before 3.1 stopped if a single bunch hit the wall
-

Algorithm Details

Basically, previous versions of tdbbu created a fixed chain of bunches and passed them around the ring, element-by-element, with each element (other than the recirculation matrix) having a 1/2 RF cycle (**clock**) step - the information was associated with **each element**.

simple view of multipass linac



FI - *fractional charge (typ. 1 or 0)*

○ Empty bunch

X - *X position*

● bunch pass#1

P_x - *X momentum*

● bunch pass#2

Y - *Y position*

P_y - *Y momentum*



Figure 3. Simple view of a multipass linac.

Even if there was no charge present in the bunch, the information was just passed from element to element on each clock step. The HOMs are represented by elements, and interact with any bunch that is "parked" at that element.

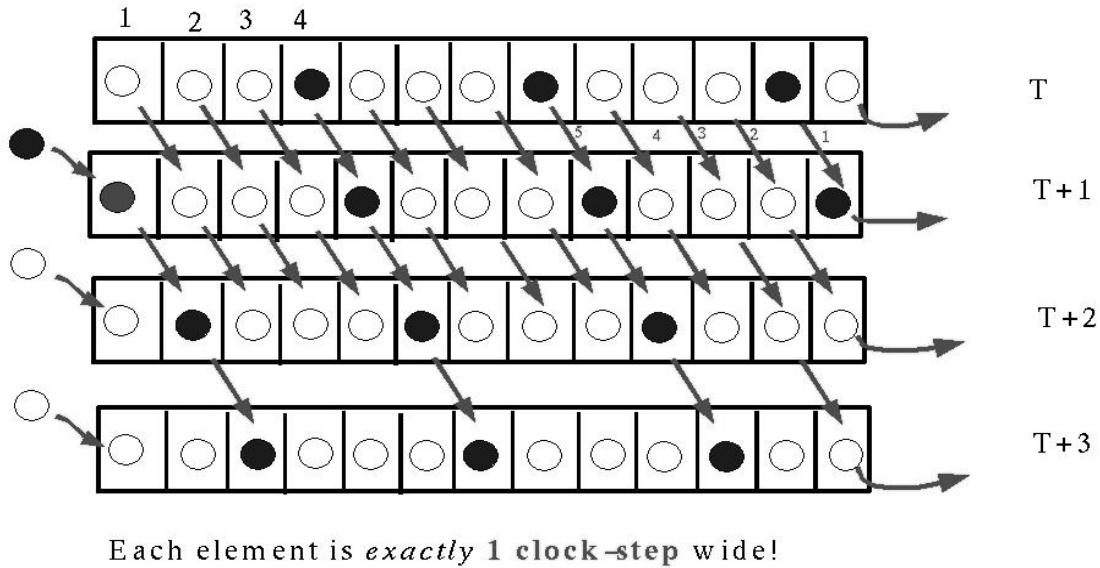


Figure 4. Cartoon of the old tdbbu algorithm.

In this version, the elements represent *operations*, and the *bunches* are stepped through the elements. At each clock step, each bunch is moved through as many elements as it can, provided that its time cannot progress up to or beyond the next master clock step. For example, in one time step a bunch may pass through many HOMs (since they represent no time delay), and multiple bunches may pass through any given HOM. At the end of each clock step, each HOM that was "visited" by a bunch is updated and its kick given to each bunch retroactively and in chronological order.

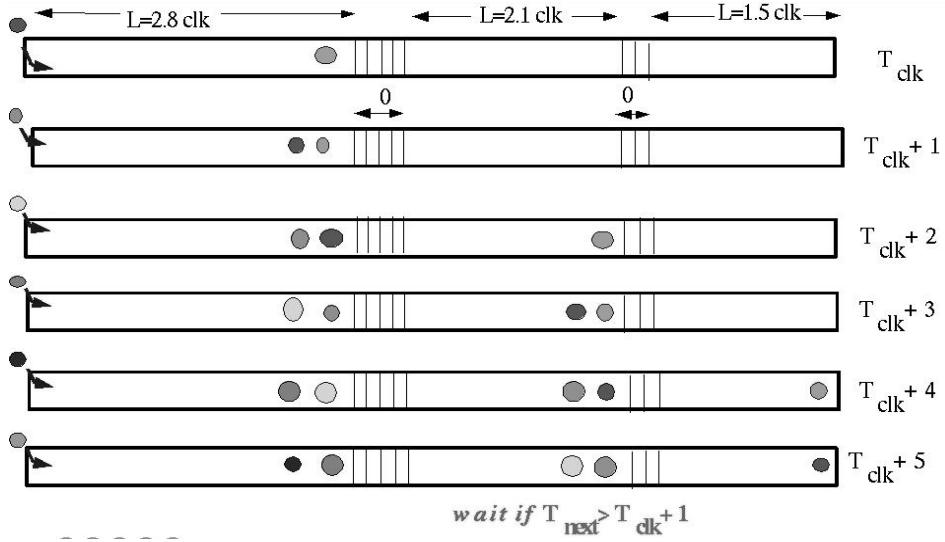


Figure 5. Cartoon of the new algorithm.

The path the bunches follow is now defined in `tdbbu_path.cmn` and filled in by routine `plan_path`; it defines a sequence of operational steps that each bunch will follow. The bunch is moved forward in time by `propagate_bunch`; this routine also predicts the time the bunch would have after the **next** element. This feature is critical: the proper chronological sequence of events at each HOM must be preserved!

Each bunch is assigned a sequential number (1,2,...), and the bunch's id# is determined by its position within a circular buffer of size **BUNCHSPACE** (currently 2000); id#s run from **0** to **BUNCHSPACE-1**. As each bunch finishes its path, information about it is recorded and the bunch removed. The absolute time (**T**) of the bunch is preserved; it is used to determine the recirculation time for each path. Each element no longer represents a fixed number of clock cycles.

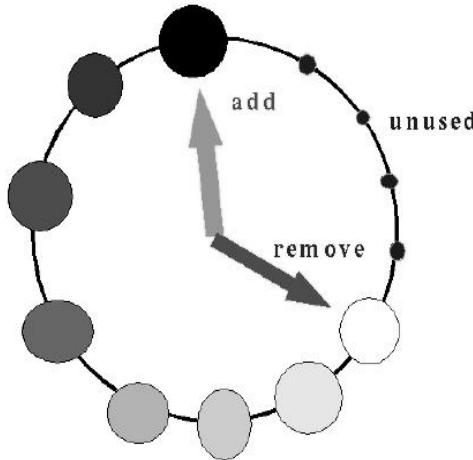


Figure 6. Method of counting bunches in `tdbbu`.

This was done to allow the bunches to have more realistic degrees of freedom - each bunch has a nominal time in uS (**T**), a time offset from the nominal time in uS (**dT**), and a momentum offset from nominal in $\delta P/P$ units (**dPP**). To make the code manageable, the whole structure of handling bunches was rewritten; the bunches have transverse position and momentum (**X**, **PX**, **Y**, **PY** in cm and MeV/c), while the **nominal** momentum is still associated with the elements.

To remain compatible with earlier versions, the momentum offset (**dP_P**), time offset (**dT_uS**), and phase offsets (**TMdegOffCrest** degrees) are all defined so the **nominal beam has zero values** for these quantities. For testing, these quantities may be suppressed using the `tdbbu_opts.par` INCLUDE file.

Typically, for each element on each pass, a **transfer matrix** (**TM**) is defined. It is in the form of a **2 x 4 matrix**, stored as a **1 x 8 array**:

$$\begin{vmatrix} X_X & X_{Px} \\ Px_X & Px_{Px} \\ Y_Y & Y_{Py} \\ Py_Y & Py_{Py} \end{vmatrix} \quad \begin{vmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ 7 & 8 \end{vmatrix}$$

For each element of type **DRIFT**, **CAVITY**, **LENSk** (LENS with k_q^2 specified) or **LENSf** (LENS with focal length specified), the **TM** matrix is "corrected" for the fractional momentum offset (δP_p) by **TM_operate**

before it is applied. For the **TM** matrices, this correction is done by routine `tdbbu_ext_TM`.

That routine also "knows" about the off-crest (or off-trough) setting of the accelerating RF for an accelerating region (a **DRIFT** with nonzero **DKE**) and the time offset of the bunch due to the **M56** and **T566** terms in the recirculation, and changes the outgoing $\delta P/P$ and time displacement dT to account for the change in phase.

The recirculation matrix has the form of a **4 x 4 matrix**, stored as a **1 x 16 array**:

$$\begin{vmatrix} X_X & X_Px & X_Y & X_Py \\ Px_X & Px_Px & Px_Y & Px_Py \\ Y_X & Y_Px & Y_Y & Y_Py \\ Py_X & Py_Px & Py_Y & Py_Py \end{vmatrix} \quad \begin{vmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{vmatrix}$$

For the **RTM** matrices, the fractional momentum offset is handled by `tdbbu_ext_RTM`; a file containing **RTM** matrices for a range of $\delta P/P$ is read in, and a linear extrapolation done between them for each element. The **RTM_operate** routine also uses the **M56** and **T566** terms, if specified, for each recirculation pass to calculate change in path length, and hence a change in arrival time, of the bunch. The time delay is that specified in the input file (as **clock cycles**) *plus* any adjustment specified on the command line for each pass: "`-T tpass#1:tpass#2:...`" (in **uS**).

Until requested at the end of a clock cycle, the **HOM_operate** routine does not update the status of each HOM: it just records the passage of the bunches, then, when requested at the end of the clock cycle, interacts with the bunches that passed by within that one clock cycle. The final state of the HOMs is also logged

Beam Loss

At each drift, lens, etc. the displacement of each bunch off-axis is checked to determine whether it exceeds the aperture times a scaling factor (`--wayayoutside N`). If it does and if the option set to stop the run immediately is set (`++Abort`), then the run finishes on the end of that clock

cycle; otherwise that bunch's charge is turned to zero and the bunch logged as lost, but still propagated until its demise. If the option (`--Abort FRACTION`) is set, the run only stops prematurely when the number of lost particles exceeds the specified fraction; then the beam is reported to be **UNSTABLE**.

Input Changes

The changes to the input file format⁶ required to represent the laser are ignored by previous versions of `tdbbu`, except for the inline comments, which are not supported by previous versions. For convenience, in version 3.0 on, in all data lines a "`!"` in column#2 or later begins a comment and is replaced with blanks before processing.

First, the location of the laser is represented by a DRIFT element, and must be identified with a "`*`" in column#1:

```
*DRIFT 1.100.0 !laser looks like a drift
```

By default, the beam dump has always been assumed to be at the end of the element preceding the recirculation matrix; that can now be changed using a "`<`" in column#1:

```
<DRIFT 1. 53.41 0.0 !beam dump
```

Also, the off-crest (or off-trough) RF settings in fields DTA4 through DTA8 for accelerating regions (DRIFTS with DTA3=dKE nonzero) in degrees, with a negative offset representing a rising voltage. Only the difference from the nominal 0 or 180 degrees is specified, so a pass#1 beam being accelerated at 10 degrees before crest and a pass#2 beam being decelerated at 170 degrees would be both described as -10 degrees. For example, the following is a 25 cm long accelerating section that adds 2.5 MeV, with -10 degrees RF phase on pass#1 (10 degrees before crest) and -10 degrees RF phase on pass#2 (10 degrees before trough):

```
1DRIFT 1. 25. 2.5 -10.0 -10. !25 cm,2.5MeV dE,-10deg...
```

A file containing the matrices or recirculation matrices (in DIMAD units) for a span of $\delta P/P$ values may be specified on the command line using "`-MF Npass Nmatrix file`" or "`-RF Npass Nmatrix file`". It is

⁶ JLAB-TN-02-043, *TDBBU and MATBBU Input File Format*, K.B.Beard, L.Merminga, B.Yunn

expected that these files are just DIMAD output.

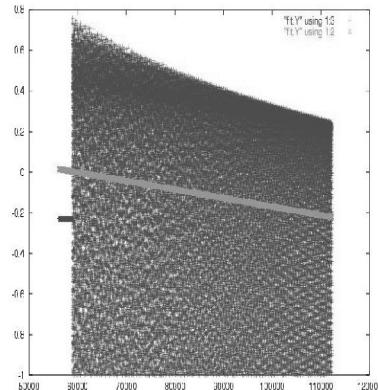
The laser-induced $\delta P/P$ momentum spread (set to be Gaussian σ for $\sigma > 0$ and uniform from $-\sigma/2$ to $+\sigma/2$ for $\sigma < 0$) is set on the command line via "+L fraction".

Then the run is the same as before; a current is specified, and the run begun.

tdbbu Hunting Algorithm

After each run at a specified current, the points from the X and Y plots are used to generate a single quantity for each in subroutine RMS2SLOPE. In the case of continuous beam, this quantity is the slope fitted to the RMS values of a moving window on the upper half of the data or MAX_plotdata points, whichever is smaller; if it is negative, the RMS values are getting smaller, if positive, they are getting larger. In the case of a duty cycle, the RMS2SLOPE is taken over all the beam ON time.

Stable:



Unstable:

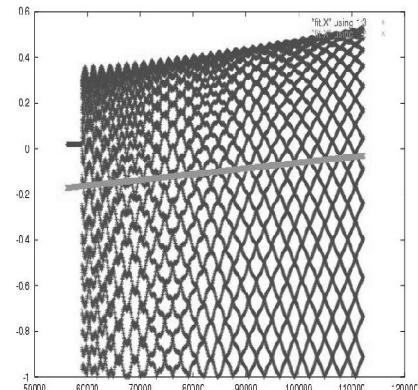


Figure 8a. Fitting the slope of the RMS value of the displacement.

The decision on whether the results are stable or not and what current to try next is made in GUESSNEXT. Each case is determined to be either STABLE [S], UNSTABLE [U], or CANT_SAY [?].

For continuous beam, there is a dead zone where it is difficult to say whether the RMS values are growing or shrinking. The default size of the dead zone is from DEADBAND(Lo) to DEADBAND(Hi) (adjustable using

--dead Lo:Hi), where the units are fractional change in normalized (to the global upper half of the data RMS) RMS/point. The default values are -5×10^{-7} and 5×10^{-7} . For a continuous beam, the beam is STABLE if the slope < DEADBAND(Lo), UNSTABLE if the slope is > DEADBAND(Hi), and CANT_SAY otherwise.

For non-continuous beam, the results are stable if the slope stays below the quantity **DutyLooksStable** (adjustable with `--looksstable setting`, default 10^{-3}).

Also, if the beam loss threshold is exceeded, both axes are declared **UNSTABLE**, irregardless of the returned slopes.

This value is kept in an internal database within subroutine **GUESSNEXT**; based on that the next current to try is chosen. The “best” region is bounded by the highest STABLE current on the low side and the lowest UNSTABLE current on the high side. The “dead” region is bounded by the lowest and highest CANT_SAY values. The low “grey” area is bounded by the highest STABLE and lowest CANT_SAY value, while the high “grey” area is bounded by the highest CANT_SAY and lowest UNSTABLE value.

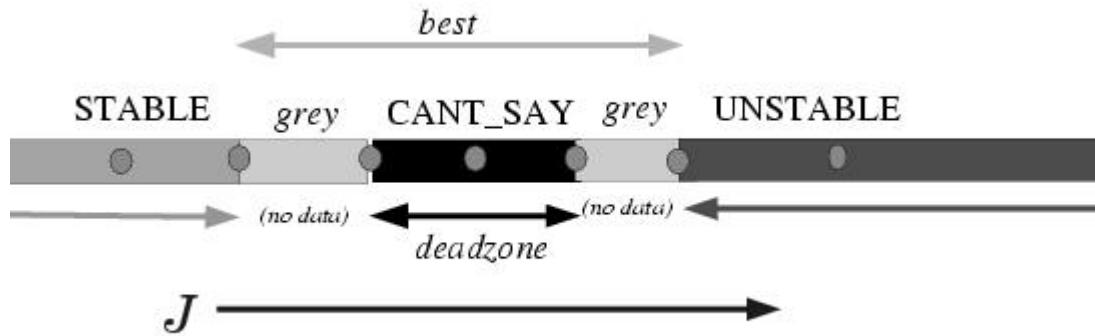


Figure 8b. Hunting algorithms' view of the current regions.

In the presence of lasing, the slope signal is “noisy”; generally, there is no practical region that appears STABLE, just a CANT_SAY and an UNSTABLE region. In that case, it is best to set the **HuntForBlowup** flag (+b) to tell the routine just minimize the high side “grey” area; otherwise, the default (-b) is to minimize both “grey” areas.

The search continues to find the "best" values until:

- (1) the best area limits are less than a CLOSE_ENOUGH_FRACTION (3% by default, but can be changed with `-c FRAC`) apart
- (2) the “grey” areas on both edges of the “dead” zone are defined to within CLOSE_ENOUGH_FRACTION/2, unless the HuntForBlowup flag is set (option `+b`), then only the high side “grey” area is considered
- (3) the maximum number of iterations have been completed (set by `-m N`)

The threshold is always expressed as a range; each iteration attempts to decrease the range. The X axis is done first, and then the Y axis, if required (unless the `-X` or `-Y` options are used).

Using tdbbu

One can get a brief list of the commonly used options:

```
$> tdbbu --help
tdbbu 3.1clid 26Jul2004, G.Kraft, B.Yunn, L.Merminga, K.Beard, TJNAF
Looks at transverse beam displacement due to HOMs in a recirculating linac

form:
$> tdbbu {option [value]} {option [value]} ... {<input>} {>output}
-h --help           - print brief list of options & quit [default]
+h ++help          - print longer list of options & quit [default]
-V --version        - print version info & quit
-v --verbose         - print informational messages
--DIMAD            - specify recirculation matrix as DIMAD (m-radian) units
--STANDARD          - specify recirculation matrix as STANDARD (cm-MeV/c) units
-i --input           FILE - input file [-]
-op --outplot        BASE - log and plot files basename (->BASE.tdlog,BASE.X,BASE.Y,BASE.gp)
-a --auto            - create plot and log filenames from input filename
-j --current          VALUES - set (list of) beam current(s) (mA*subharmonic#) to try
-J --avgcurrent       VALUE - set average beam current (mA)
-G --Guesstimate      - guess at beam current and runtime
+G ++Guesstimate     - only guess at beam current and runtime and exit
-T --Time_us           LIST - adjust recirculation time for each pass (us)
+L ++laserON          FRAC - enable laser with fractional momentum spread {<0=>flat,>0=>gaussian}
-L --laserOFF          - disable laser*
+S ++Seek             - seek thresholds automatically
--ignore              - do not abort on errors
--examples            - print some examples
--notes               - print some additional notes

also see: http://casa.jlab.org/internal/code\_library/code\_library.shtml
```

or a full list of all options from tdbbu:

```
$> tdbbu ++help
tdbbu 3.1clid 26Jul2004, G.Kraft, B.Yunn, L.Merminga, K.Beard, TJNAF
Looks at transverse beam displacement due to HOMs in a recirculating linac

form:
```

```

$> tdbbu {option [value]} {option [value]} ... { <input } { >output }
-h --help                                - print brief list of options & quit [default]
+h ++help                                 - print longer list of options & quit [default]
-V --version                               - print version info & quit
+V ++version                             - print longer version info & quit
-v --verbose                               - print informational messages
--DIMAD                                  - specify recirculation matrix as DIMAD (m-radian) units
--STANDARD                               - specify recirculation matrix as STANDARD (cm-MeV/c) units
-i --input       FILE                   - input file [-]
++DIMAD                                 - cavity matrices in DIMAD (m-radian) units
++STANDARD                            - cavity matrices in STANDARD (cm-MeV/c) units
-C --Cavmat      FILE                  - cavity description file [ ]
++DIMAD                                 - both recirculation and cavity matrices in DIMAD (m-radian) units
++STANDARD                            - both recirculation and cavity matrices in STANDARD (cm-MeV/c) units
-o --output      FILE                  - output log file [-]
+o ++output                            - no log file
-p --plot        BASE                  - plot files basename (->BASE.X,BASE.Y) [pfile]
-op --outplot     BASE                 - log and plot files basename (->BASE_tdlog,BASE.X,BASE.Y,BASE_gp)
-a --auto        FILE                  - create plot and log filenames from input filename
-r --runtime     TIME                  - set input runtime (uS)
On:Off                                - set input beam on, off dutycycle (uS:uS)
-d --dutycycle   On:Off                - set input beam on, off dutycycle (uS:uS)
+d ++dutycycle                         - force beam to be continuous
-j --current     VALUES                - set (list of) beam current(s) (mA*subharmonic#) to try
-J --avgcurrent  VALUE                 - set average beam current (mA)
-G --Guesstimate  GUESSTIMATE          - guess at beam current and runtime
+G ++Guesstimate  GUESSTIMATE          - only guess at beam current and runtime and exit
-T --Time_uS     LIST                  - adjust recirculation time for each pass (uS)
+L ++laserON    FRAC                 - enable laser with fractional momentum spread {<0=>flat,>0=>gaussian}
-L --laserOFF   FRAC                 - disable laser*
-MF --matrixfile P M FILE              - read momentum shifted DIMAD matrix for pass#P, matrix#M from FILE
-RF --recircfile P M FILE              - read momentum shifted DIMAD recirculation matrix for pass#P, matrix#M from FILE
-Lp --laserpass    N                  - set the momentum shifted recirculation pass# [1]
--M56          DESC                  - set M56 recirculation term for each matrix (DIMAD [m] units) in form pass:
-s --scanplot    BASE                 - just scan BASE.X and BASE.Y files
-S --Seek        j1...                 - seek thresholds, start w/ 1st:2nd... (mA*subharmonic#) currents
+b --blowup      FILE                  - seek edge of blowup rather than stability
-b --blowup      FILE                  - seek stability rather than edge of blowup*
+S --+Seek       Lo:Hi                - seek thresholds automatically
--dead          Lo:Hi                - set dead band limits [-0.500E-06:0.500E-06]
--fluc          CM                   - set fluctuation beneath notice [0.100E-06]
--lookstable    X                   - set normalized max RMS variation for a stable pulsed beam [0.100E-02]
--aperature     CM                   - set aperature for beam loss [0.200E+02] cm
-D --Discard     FRAC                 - initial fraction of results to discard [0.0100]
-c --closeEnough FRAC                - stop hunting when within fraction [0.0500]
-fi --flushinterval N                 - output flush interval [250000]
-m --maximum     N                 - quit after N total attempts [20]
-X --Xonly       N                 - confine seeking to X axis
-Y --Yonly       N                 - confine seeking to Y axis
--adjustprint   FILE                  - automatically adjust printing interval, if required
++adjustprint   FILE                  - do not automatically adjust printing interval*
-pa --printpath  FILE                  - print path table to stdout
+pa ++printpath  FILE                  - print path table to file FILE
-wa --wayoutside MULT                 - set the multiple of the aperature to consider a bunch lost [1.000]
+A --Abort       FILE                  - abort run if any bunch lost*
-A --Abort       FRAC                 - abort run if greater than FRAC bunches lost
--quitearly     FILE                  - allow to quit time scan early if all seems quiet
--seed          N                   - set random number seed
--limits         FILE                  - list compiled size limits and exit
--dump          ID                   - dump out complete history of sequence#ID
--ignore         FILE                  - do not abort on errors
--examples       FILE                  - print some examples
--notes          FILE                  - print some additional notes

also see: http://casa.jlab.org/internal/code\_library/code\_library.shtml

```

The input data files are shared with matbbu and are described in detail elsewhere.⁷ The changes to the input format described previously are ignored by matbbu and previous versions of tdbbu.

The only required command line option is to specify the units used in the input file (and CAVMAT file, if used). However, the **-i FILE** and either the **-op NAME** or **-a** options are almost always also used.

Unless specified otherwise on the command line, the current used is that specified on the **CMPNT** line. The current there is the harmonic (from the **BEAM** line *times* the beam current of interest. The "time to run" in the

⁷ JLAB-TN-02-043, *TDBBU and MATBBU Input File Format*, K.B.Beard, L.Merminga, B.Yunn

REF line must be set sufficiently long; the "time to turn off" must be set at least that long.

To use the values specified in the file for a single run:

```
$> tdbbu -i t11_1E6_13mA -a --DIMAD
```

The output files would be in the form of `t11_1E6_13mA.tdlog`, `t11_1E6_13mA.X`, and `t11_1E6_13mA.Y`. The latter two files just record the position of the bunches as they pass some point specified in the input file and are suitable for plotting.

There are quite a number of options, and it is perhaps easiest to go through a typical example with δP/P and lasing enabled:

```
$> tdbbu --DIMAD -C cavmat -MF 1 1 bend2irwig.out \
-RF 1 2 irwig2reinj.out -G -T -.000027 -fi 10000 \
+pa kfb10.path -i kfb.in -A 1E-3 +S +b +L -0.10 \
-op kfb10 -m 10
```

The options are handled sequentially, but the order doesn't matter unless the option changes a previous setting. Going through these options one by one:

`-+DIMAD`

The units must be specified for both the input file and any CAVMAT files used to describe cavities; this option specifies that both use DIMAD (meter-radian) units.

`-C cavmat`

The accelerating cavities are described in an external file called `cavmat`. The format is described in detail elsewhere.⁸

`-MF 1 1 bend2irwig.out`

The 1st pass, 1st matrix, is described in a DIMAD⁹¹⁰ output file named `bend2irwig.out`.

⁸ JLAB-TN-02-043, *TDBBU and MATBBU Input File Format*, K.B.Beard, L.Merminga, B.Yunn

⁹ <file:/group/casa/FEL/beard/dimad/CURRENT/DOC/index.html>

¹⁰ LAC REPORT 285 UC-28 (A), *USERS GUIDE TO THE PROGRAM DIMAD*, R. Servranckx, K.Brown, L.Schanchinger, D.Douglas, May 1985

-RF 1 2 irwig2reinj.out

The 1st pass, 2nd matrix (a recirculation matrix) is described in a DIMAD output file named `irwig2reinj.out`.

-G

In practice, it is usually convenient to allow `tddb` to guess its beginning threshold and runtime values based on the input file. Either or both of those values may be overridden by subsequent options.

-T -.000027

`tddb` returns the recirculation time based on the input file. If it is not correct, it may be corrected by setting an offset in uS for each pass (`-T uSpass1:uSpass2:...`). Typically, a very short run (`-r 3`) is done without setting `-T` and `tddb` prints out the difference in uS between what the old algorithm would have used and what the new one does use. Then that difference is specified when rerunning `tddb`.

-fi 10000

By default the output buffers to go to files are flushed at the system's discretion. The flushing interval in bunches may be given an upper limit using `-fi Nbunches`.

+pa kfb10.path

Version 3.0 creates a path for the bunches to follow; this prints out that information to a file.

-i kfb.in

This just specifies the basic input file as `kfb.in`.

-A 1E-3

This specifies that 1 bunch lost in 1000 is an acceptable rate, and will abort the run if more than that is lost.

+S

This specifies an automatic hunt for a threshold, starting with the

present current value (previously set by -G)

+b

This specifies that the hunting should concentrate on the CANT_SAY-UNSTABLE limits rather than seek the STABLE-UNSTABLE limits. When lasing, the fits to the slopes are “noisy” and it is difficult to unambiguously distinguish the STABLE from CANT_SAY condition.

+L -0.10

Here the laser is turned on (+L σ); a σ >0 gives the sigma of a Gaussian distribution in δP/P, while σ <0 specifies a uniform distribution from $-\sigma/2$ to $+\sigma/2$. Each bunch gets a δP/P as it passes through the laser element specified in the input file.

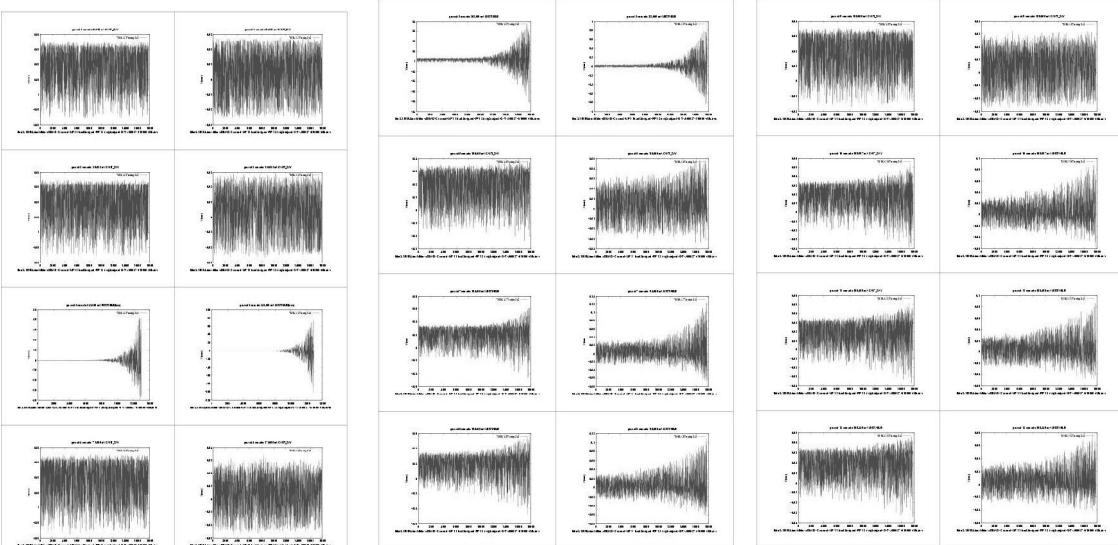
-op kfb10

All the output files will be of the form `kfb10.ext`, where a suffix of .1, .2, ... and so forth is added if multiple attempts are requested. The `name.tdlog[.N]` files contain general info on each run, while `name.X[.N]` and `name.Y[.N]` contain the bunch's positions. The `name.gp` file is a command file for plotting the results and contains summary information.

-m 12

This specifies that a maximum of 12 iterations are to be attempted. To display the results using the `name.gp` command file:

```
$> gnuplot kfb10.gp
```



```

stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=5.out-of.2669872, fractionLost=0.187E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=5.out-of.2669872, fractionLost=0.187E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=1.out-of.2669872, fractionLost=0.375E-06
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=1.out-of.2669872, fractionLost=0.375E-06
hit_wall:bunchlost=1938,total_bunches=1937693,exceedstolerance=0.10E-02, deathse#188(1938
hit_wall:bunchlost=1938,total_bunches=1937693,exceedstolerance=0.10E-02, deathse#188(1938
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=2.out-of.2669872, fractionLost=0.749E-06
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=2.out-of.2669872, fractionLost=0.749E-06
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=32.out-of.2669872, fractionLost=0.120E-04
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=32.out-of.2669872, fractionLost=0.120E-04
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=3.out-of.2669872, fractionLost=0.112E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=3.out-of.2669872, fractionLost=0.112E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=6.out-of.2669872, fractionLost=0.225E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=6.out-of.2669872, fractionLost=0.225E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=10.out-of.2669872, fractionLost=0.375E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=10.out-of.2669872, fractionLost=0.375E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=5.out-of.2669872, fractionLost=0.187E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=5.out-of.2669872, fractionLost=0.187E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=4.out-of.2669872, fractionLost=0.150E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=4.out-of.2669872, fractionLost=0.150E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=7.out-of.2669872, fractionLost=0.262E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=7.out-of.2669872, fractionLost=0.262E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=4.out-of.2669872, fractionLost=0.150E-05
stepK:(normalreturn)Time[us]=17834.812, ,bunchesLost=4.out-of.2669872, fractionLost=0.150E-05
j=188.249mA X stable:none dead:(1)60.316-(9)188.000 unstable:(12)188.249
j=188.249mA Y stable:none dead:(1)60.316-(10)188.997 unstable:(12)188.249
iteration current [mA] Xslp-Xexp-status-#pts Yslp-Yexp-status-#pts
guessshow: 1 0.60316E+02 -0.25347E-07 -0.291E-05 ? 0.25E+06 0.71308E-07 -0.310E-04 ? 0.25E+06
guessshow: 2 0.12063E+03 -0.21036E-07 0.144E-05 ? 0.25E+06 0.68203E-07 -0.209E-04 ? 0.25E+06
guessshow: 3 0.24100E+03 0.99900E+34 0.783E+02 U 0.25E+06 0.99900E+34 0.783E+02 U 0.25E+06
guessshow: 4 0.17100E+03 0.14643E-07 -0.108E-05 ? 0.25E+06 -0.30793E-07 0.151E-04 ? 0.25E+06
guessshow: 5 0.20300E+03 0.25377E-05 -0.315E-04 U 0.25E+06 0.25054E-05 -0.317E-04 U 0.25E+06
guessshow: 6 0.18600E+03 0.13387E-06 -0.461E-04 ? 0.25E+06 0.11331E-06 -0.460E-04 ? 0.25E+06
guessshow: 7 0.19400E+03 0.16078E-05 -0.373E-04 U 0.25E+06 0.13082E-05 -0.394E-04 U 0.25E+06
guessshow: 8 0.19000E+03 0.84662E-06 -0.439E-04 U 0.25E+06 0.62971E-06 -0.461E-04 U 0.25E+06
guessshow: 9 0.18800E+03 -0.26082E-07 -0.606E-05 ? 0.25E+06 -0.16478E-07 0.397E-05 ? 0.25E+06
guessshow: 10 0.18900E+03 0.63096E-06 -0.423E-04 U 0.25E+06 0.44187E-06 -0.394E-04 ? 0.25E+06
guessshow: 11 0.18850E+03 0.59535E-06 -0.491E-04 U 0.25E+06 0.34875E-06 -0.536E-04 ? 0.25E+06
guessshow: 12 0.18825E+03 0.96150E-06 -0.420E-04 U 0.25E+06 0.65131E-06 -0.471E-04 U 0.25E+06
#tdbbu: Xthr= 188.0000- 188.2488 mA Ythr= 188.9974- 188.2488 mA after 12 iterations
#tdbbu: avgXthr= 9.4000- 9.4124 mA avgYthr= 9.4499- 9.4124 mA after 12 iterations

```

Figure 9. Typical output for a given $\delta P/P$ threshold hunt.

In this example, tdbbu guessed an initial current (in average mA * subharmonic units) of 60 mA . When it found the stability condition was CANT_SAY, it then went up to 121 (CANT_SAY), and up again to 241 (UNSTABLE). It then backed down to 171 (CANT_SAY), then back up to 203 (UNSTABLE), then back down to 186 (CANT_SAY), then up to 194 (UNSTABLE), then 190 (UNSTABLE) then 188 (CANT_SAY), and so forth.

Results

Chris Tennant provided the authors a `tdbbu` input file with the best known HOMs based on his recent measurements.¹¹ That file was modified into “`kbf.in`” to give the 10kW FEL a “back leg”; the wiggler was represented by a 180 cm **DRIFT**, and the beam dump was placed just before the 100 cm preceding the first arc.

A **MATRIX** represents the first arc and a **RECIRC** the second arc. For this set of calculations, all cavities were set to 0° before crest and trough in their respective **DRIFT** statements for the first set of runs and 20° before crest and trough for the second set.

Two **DIMAD** input files, `kb5_bend2irwig.in` and `kb5_irwig2reinj.in`, were generated from Dave Douglas' general input file and used to generate the large `kb5_bend2irwig.in.out` and `kb5_irwig2reinj.in.out` output files containing the appropriate matrices with δP/P dependence.

For each δP/P setting, the average X and Y threshold limits were plotted. For both, the maximum tolerable current loss was 0.1E-4. This calculation was run on the Jlab physics cluster; the 20° case represented 156 individual runs and a total of about 4.7 CPU-days.

¹¹ C.Townsend, *private communication*

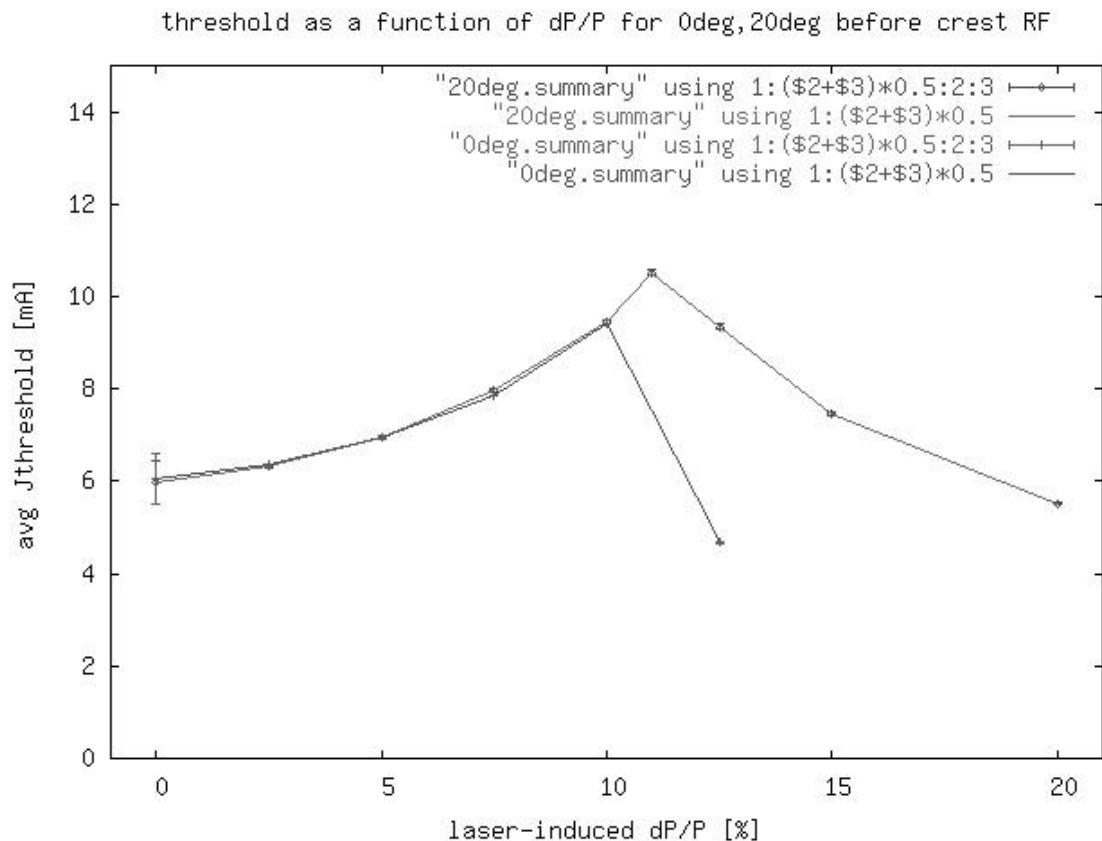


Figure 10. Average BBU current threshold as a function of $\delta P/P$ for 0° , 20° phase advance.

Other Documentation

The source code, binaries, input files, and much more additional documentation is available on the Jlab CUE system at:

/group/casa/FEL/tdbbu/3.1/

and from the Jlab FEL server:

<http://laser.jlab.org>

`tdbbu` was an old code developed with little regard for maintenance, but considerable progress has been made toward making it easier to maintain, modify and use. The current version is 3.1c1f.

A notable feature of `tdbbu` is that it uses C for the main routine (shown in lower case in Figure 11), while the bulk of the code is in FORTRAN (shown in upper case in Figure 11). This was done to allow the code to read options from the command line and to provide platform independence.

Rebuilding the code requires the author's KBB¹² and KWRAP¹³ libraries available from the same site as `tdbbu` (their purpose is to provide platform independence). It is strongly suggested that the author's organizational guide^{14 15} be read before attempting to rebuild the program, as the `Makefiles` depend on several (`KBB_*`) environmental variables to describe local idiosyncrasies. Once the files have been unpacked, the variables set for a supported platform, and the paths to the appropriate libraries set in the `Makefile`, it is only necessary to run the `Makefile`.

```
$> make help
make -f Makefile.Linux help

#
# make <command>
#
# all      - update everything
# help     - print this help
# show    - print current value of required symbols and their definitions
# clean   - delete all binaries for this platform
# distclean - delete all binaries for all platforms
```

12 KBB 7.5g Library, K.Beard,

http://casa.jlab.org/internal/code_library/casa_lib/KBB/DOC/

13 KWRAP: Kevin's Wrapper 0.1h, K.Beard,

http://casa.jlab.org/internal/code_library/casa_lib/KWRAP/DOC/

14 http://casa.jlab.org/internal/code_library/casa_lib/KBB/DOC/bs.html

15 JLAB-TN-03-001, *My Code Development Style, Organization, and Platform Dependency Guide*, K.B.Beard

```
# create_os      - create all binary directories for this platform
# destroy_os     - delete all binary directories for this platform
# destroy_all_os - delete all binary directories for all platforms
# archiveall    - create a compressed tar backup of everything
# archive       - create a compressed tar backup for export
#
make[1]: Leaving directory `/CASA/erl/BEARD/FEL/tbdbu/3.1'

$> make show
make -f Makefile.Linux show
make[1]: Entering directory `/ERL/FEL/tbdbu/1.6'
# ----- required symbols -----
printenv KBB_DEV      #- location of KBB related libraries
/home/beard/DEVELOPMENT
printenv KBB_OSTYPE    #- single keyword for current operating system
Linux
printenv KBB_CC        #- C compiler
gcc
printenv KBB_F77       #- FORTRAN compiler
g77
printenv KBB_F2C_RULE  #- suffix on F77 objects when linking C+F77
-
printenv KBB_F2C_CASE  #- name case on F77 objects when linking C+F77
LOWERCASE
# -----
make[1]: Leaving directory `/ERL/FEL/tbdbu/1.6'
```

```
$> make destroy_all_os; make create_os; make
```

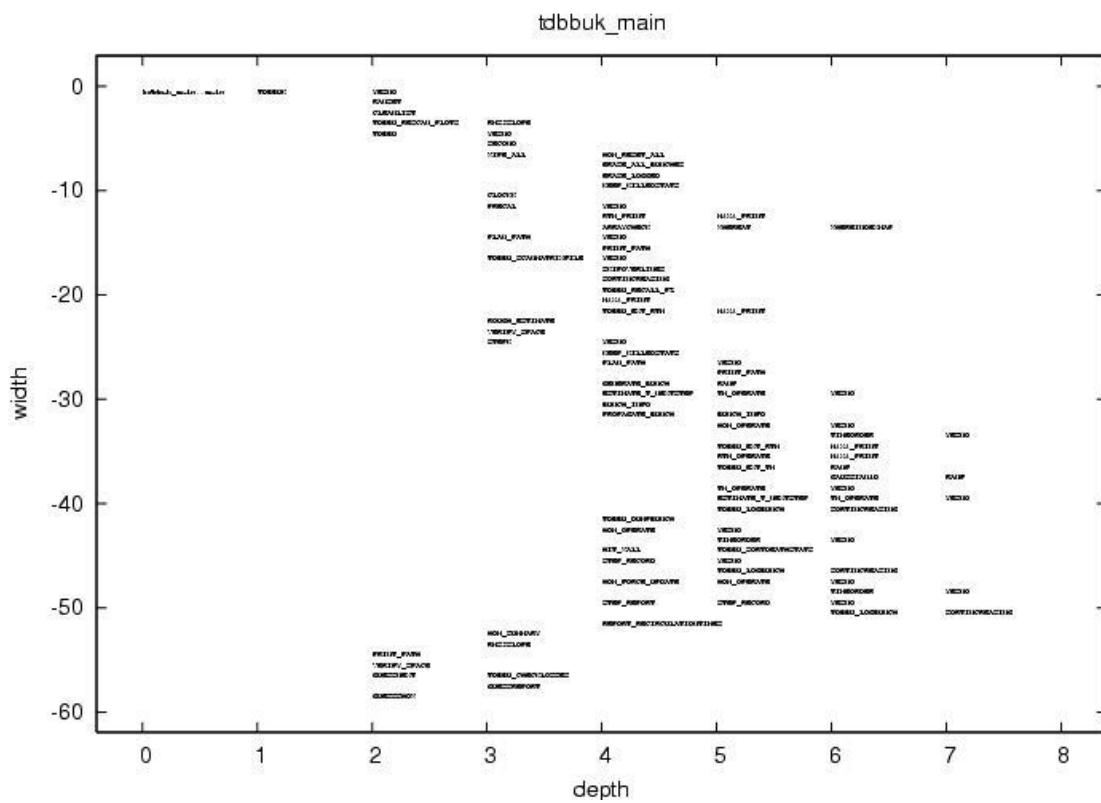


Figure 11. Dependency diagram of tdbbu generated using splitcf¹⁶.

A subroutine by subroutine description of `tdbbu` is in Appendix 3; the full hypertext version is in the online documentation.

16 http://casa.jlab.org/internal/code_library/casa_lib/SPLITCF/DOC/

Appendix 1a. tdbbu input kfb.in

```
TITLE 10kW IR FEL 145 MeV, 29jun04,w/backleg KBB
DATA
APRTR    100000.   2.0
REF      0.        1500.     1500.00   700.00   500.0    0.0
BEAM     10.0      2994.0    20.0      0.0       1.0      0.0
XPRNT    2.0       203.0     1.0
YPRNT    2.0       203.0     1.0
#CMPNT    4400.0    0.0       0.0       0.0       0.0       0.0
>DRIFT  1.100.0    0.0       !reinjection point
1DRIFT   1. 53.41   0.0
1CECAV   1. 0.0     0.0
1DRIFT   1. 5.      0.0
1CECAV   1. 0.0     0.0
1DRIFT   1. 46.06   0.0
1CECAV   1. 0.0     0.0
1DRIFT   1. 5.      0.0
1CECAV   1. 0.0     0.0
1DRIFT   1. 5.      0.0
1CECAV   1. 0.0     0.0
1DRIFT   1. 46.06   0.0
1CECAV   1. 0.0     0.0
1DRIFT   1. 5.      0.0
1CECAV   1. 0.0     0.0
1DRIFT   1. 5.      0.0
1CECAV   1. 0.0     0.0
1DRIFT   1. 53.41   0.0
1DRIFT   1. 85.1    0.0
1LENS    1.-3.597509 15.0
1DRIFT   1. 37.4    0.0
1LENS    1. 6.755323 15.0
1DRIFT   1. 37.4    0.0
1LENS    1.-3.597509 15.0
1DRIFT   1. 65.1    0.0
1DRIFT   1. 53.41   0.0
1CECAV   1. 0.0     0.0
"cavmat"
1CAVITY  29.90    2610000.  2102.607  90.0
1CAVITY  29.90    2610000.  2102.607  .0
1CAVITY  28.80    3100000.  2113.355  90.0
1CAVITY  28.80    3100000.  2113.355  .0
1DRIFT   1. 5.      0.0
1CECAV   1. 0.0     0.0
1CAVITY  29.90    6110000.  2106.007  90.0
1CAVITY  29.90    6110000.  2106.007  .0
1CAVITY  28.80    6660000.  2116.585  90.0
1CAVITY  28.80    6660000.  2116.585  .0
1DRIFT   1. 46.06   0.0
1CECAV   1. 0.0     0.0
1CAVITY  28.80    2170000.  2115.201  90.0
1CAVITY  28.80    2170000.  2115.201  .0
1DRIFT   1. 5.      0.0
1CECAV   1. 0.0     0.0
1CAVITY  29.90    1940000.  2104.683  90.0
1CAVITY  29.90    1940000.  2104.683  .0
1CAVITY  28.80    1970000.  2114.708  90.0
1CAVITY  28.80    1970000.  2114.708  .0
1DRIFT   1. 46.06   0.0
1CECAV   1. 0.0     0.0
1CAVITY  28.80    5210000.  2114.156  90.0
1CAVITY  28.80    5210000.  2114.156  .0
1DRIFT   1. 5.      0.0
1CECAV   1. 0.0     0.0
1CAVITY  29.90    2490000.  2104.201  90.0
1CAVITY  29.90    2490000.  2104.201  .0
1CAVITY  28.80    2880000.  2115.384  90.0
1CAVITY  28.80    2880000.  2115.384  .0
1CAVITY  05.20    11900000.  2123.873  90.0
1CAVITY  05.20    11900000.  2123.873  .0
1DRIFT   1. 46.06   0.0
1CECAV   1. 0.0     0.0
1DRIFT   1. 5.      0.0
1CECAV   1. 0.0     0.0
1CAVITY  29.90    2570000.  2105.565  90.0
1CAVITY  29.90    2570000.  2105.565  .0
1CAVITY  28.80    3060000.  2116.055  90.0
```

```

1CAVITY    28.80    3060000.  2116.055   .0
1DRIFT     1. 53.41    0.0
2DRIFT     1. 65.1     0.0
2LENS      1. 1.294342 15.0
2DRIFT     1. 37.4     0.0
2LENS      1. -2.550615 15.0
2DRIFT     1. 37.4     0.0
2LENS      1. 1.294342 15.0
2DRIFT     1. 85.1     0.0
1DRIFT     1. 53.41    0.0
1CECAV     1. 0.0      0.0
1DRIFT     1. 5.       0.0
1CECAV     1. 0.0      0.0
1DRIFT     1. 46.06   0.0
1CECAV     1. 0.0      0.0
1DRIFT     1. 5.       0.0
1CECAV     1. 0.0      0.0
1DRIFT     1. 46.06   0.0
1CECAV     1. 0.0      0.0
1DRIFT     1. 5.       0.0
1CECAV     1. 0.0      0.0
1DRIFT     1. 46.06   0.0
1CECAV     1. 0.0      0.0
1DRIFT     1. 5.       0.0
1CECAV     1. 0.0      0.0
<DRIFT     1. 53.41    0.0      !beam dump at end of this element
2DRIFT     1.100.0    0.0
    MATRIX          !1st arc
*DRIFT      180.0      !laser wiggler
$RECIRC 1.          !2nd arc
$CALC      0.
0.1,0.,0,0.0,0.,0
0.1,0.,0,0.0,0.,0
386
1.,0,0,0
0,1.,0,0
0,0,1.,0
0,0,0,1.
572
0.443548  -18.14799   0.0 0.0
0.0410076  0.576699    0.0 0.0
0.0 0.0    -1.152552   17.68565
0.0 0.0    0.03581095  -1.417150
0.,0.,0.,0.,0.,
0.,0.,0.,0.,0.,
2222
0,0,0,0
0,0,0,0
0,0,0,0
0,0,0,0

```

Appendix 1b. tdbbu input cavmat

```

! ZONE 2 ACCELERATION
5.075790532000000 70. 0.778246 0.561123 -0.101322 0.783395 0.778246 0.561123 -0.101322 0.783395
5.075790532000000 70. 0.846077 0.600681 -0.054275 0.847917 0.846077 0.600681 -0.054275 0.847917
5.075790532000000 70. 0.882194 0.622667 -0.033786 0.883019 0.882194 0.622667 -0.033786 0.883019
5.075790532000000 70. 0.904606 0.636675 -0.023047 0.905029 0.904606 0.636675 -0.023047 0.905029
5.075790532000000 70. 0.919863 0.646383 -0.016723 0.920100 0.919863 0.646383 -0.016723 0.920100
5.075790532000000 70. 0.930917 0.653509 -0.012686 0.931057 0.930917 0.653509 -0.012686 0.931057
5.075790532000000 70. 0.939294 0.658962 -0.009953 0.939380 0.939294 0.658962 -0.009953 0.939380
5.075790532000000 70. 0.945860 0.663270 -0.008017 0.945915 0.945860 0.663270 -0.008017 0.945915

! ZONE 3 ACCELERATION
6.980883460000000 70. 0.933387 0.655120 -0.011875 0.933524 0.933387 0.655120 -0.011875 0.933524
6.980883460000000 70. 0.941211 0.660223 -0.009386 0.941297 0.941211 0.660223 -0.009386 0.941297
6.980883460000000 70. 0.947391 0.664283 -0.007605 0.947447 0.947391 0.664283 -0.007605 0.947447
6.980883460000000 70. 0.952396 0.667591 -0.006287 0.952433 0.952396 0.667591 -0.006287 0.952433
6.980883460000000 70. 0.956532 0.670338 -0.005284 0.956556 0.956532 0.670338 -0.005284 0.956556
6.980883460000000 70. 0.960007 0.672656 -0.004504 0.960023 0.960007 0.672656 -0.004504 0.960023
6.980883460000000 70. 0.962968 0.674637 -0.003884 0.962978 0.962968 0.674637 -0.003884 0.962978
6.980883460000000 70. 0.965521 0.676351 -0.003384 0.965527 0.965521 0.676351 -0.003384 0.965527

! ZONE 4 ACCELERATION
5.077089683000000 70. 0.976445 0.683742 -0.001613 0.976442 0.976445 0.683742 -0.001613 0.976442
5.077089683000000 70. 0.977505 0.684463 -0.001474 0.977502 0.977505 0.684463 -0.001474 0.977502
5.077089683000000 70. 0.978473 0.685124 -0.001353 0.978470 0.978473 0.685124 -0.001353 0.978470
5.077089683000000 70. 0.979362 0.685730 -0.001246 0.979358 0.979362 0.685730 -0.001246 0.979358
5.077089683000000 70. 0.980180 0.686289 -0.001151 0.980176 0.980180 0.686289 -0.001151 0.980176
5.077089683000000 70. 0.980935 0.686806 -0.001066 0.980932 0.980935 0.686806 -0.001066 0.980932
5.077089683000000 70. 0.981635 0.687286 -0.000991 0.981632 0.981635 0.687286 -0.000991 0.981632
5.077089683000000 70. 0.982286 0.687731 -0.000923 0.982283 0.982286 0.687731 -0.000923 0.982283

! ZONE 2 DECELERATION
-5.075790532000000 70. 1.017367 0.712290 -0.000958 1.017354 1.017367 0.712290 -0.000958 1.017354
-5.075790532000000 70. 1.017992 0.712737 -0.001029 1.017977 1.017992 0.712737 -0.001029 1.017977
-5.075790532000000 70. 1.018664 0.713218 -0.001109 1.018647 1.018664 0.713218 -0.001109 1.018647
-5.075790532000000 70. 1.019387 0.713737 -0.001199 1.019369 1.019387 0.713737 -0.001199 1.019369
-5.075790532000000 70. 1.020169 0.714298 -0.001300 1.020149 1.020169 0.714298 -0.001300 1.020149
-5.075790532000000 70. 1.021017 0.714907 -0.001414 1.020995 1.021017 0.714907 -0.001414 1.020995
-5.075790532000000 70. 1.021939 0.715570 -0.001544 1.021915 1.021939 0.715570 -0.001544 1.021915
-5.075790532000000 70. 1.022945 0.716295 -0.001693 1.022918 1.022945 0.716295 -0.001693 1.022918

! ZONE 3 DECELERATION
-6.980883460000000 70. 1.033202 0.723736 -0.003626 1.033130 1.033202 0.723736 -0.003626 1.033130
-6.980883460000000 70. 1.035562 0.725463 -0.004182 1.035475 1.035562 0.725463 -0.004182 1.035475
-6.980883460000000 70. 1.038283 0.727461 -0.004877 1.038178 1.038283 0.727461 -0.004877 1.038178
-6.980883460000000 70. 1.041455 0.729799 -0.005762 1.041324 1.041455 0.729799 -0.005762 1.041324
-6.980883460000000 70. 1.045199 0.732572 -0.006910 1.045034 1.045199 0.732572 -0.006910 1.045034
-6.980883460000000 70. 1.049686 0.735915 -0.008440 1.049471 1.049686 0.735915 -0.008440 1.049471
-6.980883460000000 70. 1.055159 0.740023 -0.010541 1.054871 1.055159 0.740023 -0.010541 1.054871
-6.980883460000000 70. 1.061985 0.745191 -0.013538 1.061583 1.061985 0.745191 -0.013538 1.061583

! ZONE 4 DECELERATION
-5.077089683000000 70. 1.051058 0.736947 -0.008961 1.050834 1.051058 0.736947 -0.008961 1.050834
-5.077089683000000 70. 1.056855 0.741309 -0.011272 1.056551 1.056855 0.741309 -0.011272 1.056551
-5.077089683000000 70. 1.064131 0.746837 -0.014608 1.063703 1.064131 0.746837 -0.014608 1.063703
-5.077089683000000 70. 1.073526 0.754075 -0.019680 1.072897 1.073526 0.754075 -0.019680 1.072897
-5.077089683000000 70. 1.086155 0.763960 -0.027942 1.085128 1.086155 0.763960 -0.027942 1.085128
-5.077089683000000 70. 1.103951 0.778266 -0.042765 1.102116 1.103951 0.778266 -0.042765 1.102116
-5.077089683000000 70. 1.130848 0.800813 -0.073540 1.126976 1.130848 0.800813 -0.073540 1.126976
-5.077089683000000 70. 1.175684 0.841568 -0.155449 1.164771 1.175684 0.841568 -0.155449 1.164771

```

Appendix 2a. DIMAD input kb5_bend2irwig.in

```

utransport
TITLE
IR Upgrade Design Revision 1.1.2, 8 October 2001

!! COMMENTS FOR REV 1.0
!! fint numbers approximately those provided by Jeff Karn
!! 32 m/16 m optical cavities for 75 MHz drive laser
!! minor interference of end-can with fat chic
!! little chic dipoles approximately DW's
!! path length 10th subharmonic/drive laser 20th subharmonic
!! so that accelerated/energy recovered bunches essentially
!! interleaved

! COMMENTS FOR REV 1.1-----
!
! injection momentum: 10 MeV/c
! driver dynamic range: 80-210 MeV/c
! baseline set point: 145 MeV/c
! note these are momenta!
!
! 1. injection line changed to sector dipoles as in IR Demo
! per B. Yunn specifications (e-mail of 11/14/2000)
! layout evaluated in "rev 1.1 layouts.xls"
!
! 2. linac accelerates from 10 MeV/c to 145 MeV/c
! approximate module gains: 40 MeV, 55 MeV, 40 MeV
!
! 3. linac slots allow use of 7-cell modules in all locations
! with 1 m (center to center) triplets in warm regions

!
! comments for rev 1.12
!
! 1. widened linac triplets by 2.4 cm
! 2. moved er dump chicane 0.25 m downstream
! 3. moved er line triplet 0.25 m downstream
! (this is not in this file - it is in the dump line file)
! 4. initial quad strengths from rev111 output

pi=3.14159265358979
cdtor=pi/180
crtod=180/pi
rflambda=0.20026216299

!DRIFTS
DFAC: DRIFT, L = 8.2500

d0:drift, l=1
! adjust to set path length - delta in this gives 4*delta on pathlength:
!d0a:drift, l=0.43429490824
d0a:drift, l=0.5 + 0.0208744373 - rflambda/2 + 0.00731255025
dbcm: drift, l=(2*rflambda+0.25)

DF1: DRIFT, L=0.875 - 0.024
DF2: DRIFT, L=0.35 + 0.024
DF3: DRIFT, L=0.675 - 0.024

DA1: DRIFT, L=0.5341
DA2: DRIFT, L=0.0500
DA3: DRIFT, L=0.4606

! optics solution from "upgrade linac optics 16 November 2000 alt rev112.xls"
!21.65441457 -0.837499584 16.65441457 0.837499584 1 21.4234518 -0.956381413 17.4234518
0.956381413
KQ21 =-3.597509424
KQ31 = 6.755323252
KQ41 = 1.29434205
KQ51 =-2.550615217

KQ22 =KQ21 * 50/105.0102458
KQ32 =KQ31 * 50/105.0102458
KQ42 =KQ41 * 105/ 50.01035005
KQ52 =KQ51 * 105/ 50.01035005

Q21: QUADRUPOLE, L=lqm/6, K1=KQ21, aperture=0.0375
Q31: QUADRUPOLE, L=lqm/6, K1=KQ31, aperture=0.0375
Q41: QUADRUPOLE, L=lqm/6, K1=KQ41, aperture=0.0375
Q51: QUADRUPOLE, L=lqm/6, K1=KQ51, aperture=0.0375

Q22: QUADRUPOLE, L=lqm/6, K1=KQ22, aperture=0.0375
Q32: QUADRUPOLE, L=lqm/6, K1=KQ32, aperture=0.0375
Q42: QUADRUPOLE, L=lqm/6, K1=KQ42, aperture=0.0375
Q52: QUADRUPOLE, L=lqm/6, K1=KQ52, aperture=0.0375

!POSITION MONITORS
PMA:MONITOR,L= 0.254
PM:MONITOR,L= 0

! ACCELERATING SECTIONS
!----- 1ST PASS - UP -----
AC001:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC002:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC003:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC004:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC005:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10

```

```

AC006:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=-10
AC007:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=-10
AC008:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=-10
AC009:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC010:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC011:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC012:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC013:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC014:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC015:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC016:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC017:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC018:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC019:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC020:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC021:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC022:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC023:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC024:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
!----- 2ND PASS - DOWN -----
AC025:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC026:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC027:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC028:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC029:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC030:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC031:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC032:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC033:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC034:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC035:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC036:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC037:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC038:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC039:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC040:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC041:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC042:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC043:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC044:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC045:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC046:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC047:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC048:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170

!LATTICE
CELLA(ACA,ACB): LINE=(ACA,DA2,ACB)
SCRYO:LINE=( DA1,CELLA(AC1,AC2),DA1)
CRYO(ACA,ACB,ACC,ACD,ACE,ACF,ACG,ACH):LINE=(DA1,&
CELLA(ACA,ACB),DA3,CELLA(ACC,ACD),DA3,&
CELLA(ACE,ACF),DA3,CELLA(ACG,ACH),DA1)

warm(q1,q2,q3): line=(df1, 6*q1, df2, 6*q2, df2, 6*q3, df5)
mraw(q1,q2,q3): line=(df3, 6*q1, df2, 6*q2, df2, 6*q3, df1)

LINAC1: LINE=(d0, &
CRYO(AC001,AC002,AC003,AC004,AC005,AC006,AC007,AC008),&
warm(q21, q31, q21), &
CRYO(AC009,AC010,AC011,AC012,AC013,AC014,AC015,AC016),&
mraw(q41, q51, q41), &
CRYO(AC017,AC018,AC019,AC020,AC021,AC022,AC023,AC024),&
d0)

LINAC2: LINE=(d0, &
CRYO(AC025,AC026,AC027,AC028,AC029,AC030,AC031,AC032),&
warm(q22, q32, q22), &
CRYO(AC033,AC034,AC035,AC036,AC037,AC038,AC039,AC040),&
mraw(q42, q52, q42), &
CRYO(AC041,AC042,AC043,AC044,AC045,AC046,AC047,AC048),&
d0)

arcfit1:marker
arcfit2:marker
!21.65441457      -0.837499584     16.65441457      0.837499584     1      21.4234518      -0.956381413     17.4234518
0.956381413
beta0x = 21.65441457
alpha0x = -0.837499584
beta0y = 21.42345187
alpha0y = -0.956381413

beta0x = 16.65441457
alpha0x = 0.837499584
beta0y = 17.42345187
alpha0y = 0.956381413
momcompt= 0.19205

arcmat: matrix, r11=sqrt(beta0x/beta0x), &
r12=0, &
r21=(alpha0x-alpha0x)/sqrt(beta0x*beta0x), &
r22=sqrt(beta0x/beta0x), &
r33=sqrt(beta0y/beta0y), &
r34=0, &
r43=(alpha0y-alpha0y)/sqrt(beta0y*beta0y), &
r44=sqrt(beta0y/beta0y), &
r55=momcompt, r56=0, r66=1

alllinac: line=(linac1,arcfit1,arcmat,arcfit2,linac2)

```

```

! arc data -----
injslot: drift, l=2.98963
! Excel value: injface=3.867235220330010
! dimad value:
injface=.38678003679847E+01

breinj1: sbend, l=0.217416041076069, angle=1.431841557712840, &
e1=0, e2=(1.431841557712840-20), tilt=180, &
hgap=0.0375, fint=0.3, hgpx=0.0375, fintx=0.3
breinj2: sbend, l=0.217416041076069, angle=1.431841557712840, &
e2=0, el=injface, &
hgap=0.0375, fint=0.3, hgpx=0,
breinj3: sbend, l=0.217416041076069, angle=1.431841557712840, &
e1=0, e2=injface, &
hgap=0, fint=0, hgpx=0.0375, fintx=0.3
breinj4: sbend, l=0.217416041076069, angle=1.431841557712840, &
e2=0, el=(1.431841557712840-20), tilt=180, &
hgap=0.0375, fint=0.3, hgpx=0.0375, fintx=0.3

dreinj1: drift, l=0.815467800213433
dreinj2: drift, l=0.5
dreinj3: drift, l=0.48963

reinj: line=(breinj1, dreinj2, &
breinj3, dreinj1, breinj4, dreinj3)

bext1: sbend, l=0.217416041076069, angle=1.431841557712840, &
e1=0, e2=(1.431841557712840-20), &
hgap=0.0375, fint=0.3, hgpx=0.0375, fintx=0.3
bext2: sbend, l=0.217416041076069, angle=1.431841557712840, &
e2=0, el=injface, tilt=180, &
hgap=0.0375, fint=0.3, hgpx=0,
bext3: sbend, l=0.217416041076069, angle=1.431841557712840, &
e1=0, e2=injface, tilt=180, &
hgap=0, fint=0, hgpx=0.0375, fintx=0.3
bext4: sbend, l=0.217416041076069, angle=1.431841557712840, &
e2=0, el=(1.431841557712840-20), &
hgap=0.0375, fint=0.3, hgpx=0.0375, fintx=0.3

extra: line=(dbcm, &
bext1, dreinj1, bext2, &
bext3, dreinj1, bext4)

lmatch1 = 6 + 1 + 0.5 - 1
lmatch2 = 17.639630 + 2 + 2*0.075904533259 + 1 - 0.2149809 &
+ 0.0631718 - 2 + 2*rflambda

lqm = 0.15

! match to arc -----
kqm11 =-.29161950608304E+01
!kqm12 = .64274479269415B+01
kqm12 = 6.5
kqm13 =-.45715119232089E+01
kqm14 = .209164721219162E+01
kqm15 =-.30425614550167E+01
kqm16 = 6

qml1: quadrupole, l=lqm/6, k1=kqm11
qml2: quadrupole, l=lqm/6, k1=kqm12
qml3: quadrupole, l=lqm/6, k1=kqm13
qml4: quadrupole, l=lqm/6, k1=kqm14
qml5: quadrupole, l=lqm/6, k1=kqm15
qml6: quadrupole, l=lqm/6, k1=kqm16

ldm11=((lmatch1/8)-(lqm/2)) + 0.2
ldm12=(0.75-lqm)
ldm13=(lmatch1/8)-lqm - 0.2

dm11alt: drift, l=(ldm11-0.25)
dm11: drift, l=ldm11
dm12: drift, l=ldm12
dm13: drift, l=ldm13

match1: line=(dm11alt, 6*qml1, dml3, 6*qml2, dm13, 6*qml3, &
dml1, dml1, &
6*qml4, dml3, 6*qml5, dm13, 6*qml6, dm11)

! match to wiggler garden-----
kqm21= -4.4 *1.125
kqm22= 4.4
kqm23= -4.4
kqm24= 4.4
kqm25= -4.4
kqm26= 4.4
kqm27= -4.4
kqm28= 4.4
kqm29= -4.4
kqm210= 4.4*0.75
kqm211= -4.4*0.5
kqm212= 4.4*0.25

qm21: quadrupole, l=lqm/6, k1=kqm21
qm22: quadrupole, l=lqm/6, k1=kqm22
qm23: quadrupole, l=lqm/6, k1=kqm23
qm24: quadrupole, l=lqm/6, k1=kqm24
qm25: quadrupole, l=lqm/6, k1=kqm25
qm26: quadrupole, l=lqm/6, k1=kqm26
qm27: quadrupole, l=lqm/6, k1=kqm27
qm28: quadrupole, l=lqm/6, k1=kqm28
qm29: quadrupole, l=lqm/6, k1=kqm29
qm210:quadrupole, l=lqm/6, k1=kqm210
qm211:quadrupole, l=lqm/6, k1=kqm211
qm212:quadrupole, l=lqm/6, k1=kqm212

```

```

ldm21=((lmatch2/12)-lqm)/2
ldm23=(lmatch2/12)-lqm

dm21: drift, l=ldm21
dm23: drift, l=ldm23

blcell: line=(3*qm26,dm23,6*qm27,dm23,3*qm26)

fm21: marker
fm22: marker
fm2c1: marker
fm2c2: marker

match2: line=(dm21, 6*qm21, dm23, &
             6*qm22, dm23, &
             6*qm23, dm23, &
             6*qm24, dm23, &
             6*qm25, dm23, &
             3*qm26, fm21, 2*qm26, dm23, &
             3*qm27, 3*qm27, dm23, &
             3*qm28, fm22, 3*qm28, dm23, &
             6*qm29, dm23, &
             6*qm210,dm23, &
             6*qm211,dm23, &
             6*qm212,dm21)

! match to ir wiggler -----
kqm31 = .29002763186331E+01
kqm32 = -.33174132988901E+01
kqm33 = 0
kqm34 = 0
kqm35 = .38024948331400E+01
kqm36 = -.34839913449125E+01

qm31: quadrupole, l=lqm/6, k1=kqm31
qm32: quadrupole, l=lqm/6, k1=kqm32
qm33: quadrupole, l=lqm/6, k1=kqm33
qm34: quadrupole, l=lqm/6, k1=kqm34
qm35: quadrupole, l=lqm/6, k1=kqm35
qm36: quadrupole, l=lqm/6, k1=kqm36

lmatch3=10

!ldm31=0.925
!ldm32=0.85
!ldm33=0.425
ldm31 =(lmatch3-4)/4 - lqm/2 + 0.2
ldm31a=(lmatch3-4)/4 - lqm/2 + 0.2 + 0.14
ldm32= 1.0 - lqm - 0.2

dm31: drift, l=ldm31
dm31a: drift, l=ldm31a
dm32: drift, l=ldm32
!dm33: drift, l=ldm33

bump: drift, l=3

match3: line=(dm31, 6*qm31, dm32, 6*qm32, dm32, 6*qm33, dm31, &
              dm31, 6*qm34, dm32, 6*qm35, dm32, 6*qm36, dm31a)

! match from wiggler garden -----
!kQM51 = .50285990019186E+01
!kQM52 = -.41521663367068E+01
!kQM53 = .18066117437057E+00
!kQM54 = .49211192973563E+01
!kQM55 = -.38723341038288E+01
!kQM56 = .30319519805250E+00
!kQM51 = -2
!kQM52 = 4
!kQM53 = -2
!kQM54 = 0
!kQM55 = 3
!kQM56 = -3

kQM51 = -.12146182906751E+01
kQM52 = -.63253855991718E+01
kQM53 = .54766661899922E+01
kQM54 = -.35326532555102E+01
kQM55 = .17078204392578E+00
kQM56 = .18644842262609E+01

qm51: quadrupole, l=lqm/6, k1=kqm51
qm52: quadrupole, l=lqm/6, k1=kqm52
qm53: quadrupole, l=lqm/6, k1=kqm53
qm54: quadrupole, l=lqm/6, k1=kqm54
qm55: quadrupole, l=lqm/6, k1=kqm55
qm56: quadrupole, l=lqm/6, k1=kqm56

lmatch5=10

!ldm51=1.680637
!ldm52=1.322181
!ldm53=0.15 + 0.075
ldm51=0.5 0.14 + 0.24
!ldm52=1.292761222056
ldm52=1.25 - 0.15 - 0.15 - 0.1
ldm53=1.38947755588799 + (1.292761222056-1.25)*2 + 2*0.15 + 0.1 &
      - 0.12
ldm54=0.5 + 0.15 + 4*0.15 + 2*0.1

dm51: drift, l=ldm51
dm52: drift, l=ldm52
dm53: drift, l=ldm53
dm54: drift, l=ldm54

```

```

!ldm52b=0.45171286141/cos(cdtor*7.5)
!ldm52c=0.25
!ldm52b: drift, l=ldm52b
!ldm52c: drift, l=ldm52c

match5: line=(-(dm54, 6*qm51, dm52, 6*qm52, dm52, 6*qm53, dm53, &
dm53, 6*qm54, dm52, 6*qm55, dm52, 6*qm56, dm51))
!
      bridging from optical cavity chicane to endloop
!
      match to reinjection -----
kQM61 = -.82564998314733E+00
!kQM62 = -.45261959060859E+01
kQM62 = -5
kQM63 = -.10826098940769E+01
kQM64 = 0
kQM65 = .41236768158546E+01
kQM66 = -.38640794706814E+01

qm61: quadrupole, l=lqm/6, k1=kqm61
qm62: quadrupole, l=lqm/6, k1=kqm62
qm63: quadrupole, l=lqm/6, k1=kqm63
qm64: quadrupole, l=lqm/6, k1=kqm64
qm65: quadrupole, l=lqm/6, k1=kqm65
qm66: quadrupole, l=lqm/6, k1=kqm66

!ldm61=((lmatch1/8)-(lqm/2))
!ldm62=(0.75-lqm)
!ldm63=(lmatch1/8)-lqm

!ldm61=0.416624
!ldm62=1.179402
!ldm63=0.157883 + 0.075
ldm61=0.5
ldm62=0.9875
ldm63=0.5+0.075

dm61: drift, l=ldm61
dm62: drift, l=ldm62
dm63: drift, l=ldm63

match6: line=(       6*qm61, dm62, 6*qm62, dm62, 6*qm63, &
dm63, dm63, &
6*qm64, dm62, 6*qm65, dm62, 6*qm66, dm61)

!
      endloop data -----
rho=1
rhop=1.2
rhopp=1.2
bang=43.383565*pi/180
facel= 2
face2=16.5
face3= 0

bla: sbend, l=rhop*bang/10, angle=bang*crtod/10, &
e1=face1, e2=0, &
hgap=0.0375, fint=0.26, hgpx=0, fintx=0
blb: sbend, l=rhop*bang/10, angle=bang*crtod/10, &
e1=0, e2=0
blc: sbend, l=rhop*bang/10, angle=bang*crtod/10, &
e1=0, e2=face1, &
hgap=0, fint=0, hgpx=0.0375, fintx=0.26

b2a: sbend, l=rhopp*bang/10, angle=bang*crtod/10, &
e1=face2, e2=0, tilt=180, &
hgap=0.0375, fint=0.26, hgpx=0, fintx=0
b2b: sbend, l=rhopp*bang/10, angle=bang*crtod/10, &
e1=0, e2=0, tilt=180
b2c: sbend, l=rhopp*bang/10, angle=bang*crtod/10, &
e1=0, e2=face2, tilt=180, &
hgap=0, fint=0, hgpx=0.0375, fintx=0.26

b1: line=(bla, 8*b1b, b1c)
b2: line=(b2a, 8*b2b, b2c)

b1b: sbend, l=rho*pi/60, angle=pi*crtod/60, el= face3, e2=0,&
hgap=0.0375, fint=0.26, hgpx=0, fintx=0
bb2: sbend, l=rho*pi/60, angle=pi*crtod/60, el=0, e2=0
bb3: sbend, l=rho*pi/60, angle=pi*crtod/60, el=0, e2=0
bb4: sbend, l=rho*pi/60, angle=pi*crtod/60, el=0, e2=0
bb5: sbend, l=rho*pi/60, angle=pi*crtod/60, el=0, e2= face3,&
hgap=0, fint=0, hgpx=0.0375, fintx=0.26

ldell1 =0.28514265396274E+00
ldella=0.25
ldellb=0.075
ldel12 =0.29846876655049E+00
ldel12a=0.25
ldel12b=0.075

dell1: drift, l=ldell1
della: drift, l=ldella
dellb: drift, l=ldellb
del2: drift, l=ldel12
del2a: drift, l=ldel12a
del2b: drift, l=ldel12b

kqt11 = .11887968731297E-01
kqt12 =-.80401660588917E-01
kqt21 =-.20510638424916E-02
kqt22 = .13813705765973E-01

qt11a: quadrupole, l=0.15/6, k1=kqt11
qt11b: quadrupole, l=0.15/6, k1=kqt11
qt12a: quadrupole, l=0.15/6, k1=kqt12

```

```

qt12b: quadrupole, l=0.15/6, k1=kqt12
qt21a: quadrupole, l=0.15/6, k1=kqt21
qt21b: quadrupole, l=0.15/6, k1=kqt21
qt22a: quadrupole, l=0.15/6, k1=kqt22
qt22b: quadrupole, l=0.15/6, k1=kqt22

ks11a =-.668177084714068+01
ks12a = .106485278796218+02
ks12b = .114313679873578+02
ks11b =-.87268253610551E+01
ks21a =-.667768306014532+01
ks22a = .928139422526318+01
ks22b = .92871457140399E+01
ks21b =-.669340003258208+01

s11a: sextupole, l=0.15/6, k2=ks11a
s11b: sextupole, l=0.15/6, k2=ks11b
s12a: sextupole, l=0.15/6, k2=ks12a
s12b: sextupole, l=0.15/6, k2=ks12b
s21a: sextupole, l=0.15/6, k2=ks21a
s21b: sextupole, l=0.15/6, k2=ks21b
s22a: sextupole, l=0.15/6, k2=ks22a
s22b: sextupole, l=0.15/6, k2=ks22b

fitbb: marker

endloop1: line=(d0a, b1, dell1, (2*dell1b), della, (6*s11a, della, 6*qt11a), &
               dell1, b2, &
               del2, 2*del2b, del2a, 6*s12a, del2a, 6*qt12a, del2, &
               bbl, 14*bb2, 15*bb3, fitbb, 15*bb3, 14*bb4, bb5,&
               del2, 6*qt12b, del2a, 6*s12b, del2a, 2*del2b, del2, &
               b2, dell1, (6*qt11b,della, 6*s11b), della, (2*dell1b), dell1, b1, d0a)

oo: multipole, l=0, k3=-1.7
!oo: multipole, l=0, k3=0

endloop2: line=(d0a, b1, dell1, (2*dell1b), della, (6*s21a, della, 6*qt21a), &
               dell1, b2, &
               del2, del2b, oo, del2b, del2a, 6*s22a, del2a, 6*qt22a, del2, &
               bbl, 14*bb2, 15*bb3, fitbb, 15*bb3, 14*bb4, bb5,&
               del2, 6*qt22b, del2a, 6*s22b, del2a, del2b, oo, del2b, del2, &
               b2, dell1, (6*qt21b,della, 6*s21b), della, (2*dell1b), dell1, b1, d0a)

fit1:marker
fit2:marker
fit3:marker
fit4:marker
fit5:marker
fit6:marker
fit7:marker

! optical cavity chicane

bangx=23.13749962
facex=-18.24606538
bangc=20.24606538*cdtor
!(43.whatever-bangx)*cdtor

rhoc=1.2

ldch1=0
ldch2=1.779414643-ldch1/cos(bangc)
ldch3=0.5

dch1: drift, l=ldch1
dch2: drift, l=ldch2
dch3: drift, l=ldch3

bch1a: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=bangc*crtod/2, e2=0, &
        hgap=0.0375, fint=0.26, hgapx=0, fintx=0
bch1b: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=0,&
        hgap=0, fint=0, hgapx=0, fintx=0
bch1c: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=bangc*crtod/2,&
        hgap=0, fint=0, hgapx=0.0375, fintx=0.26

bch2a: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=bangc*crtod/2, e2=0, tilt=180,&
        hgap=0.0375, fint=0.26, hgapx=0, fintx=0
bch2b: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=0, tilt=180,&
        hgap=0, fint=0, hgapx=0, fintx=0
bch2c: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=bangc*crtod/2, tilt=180,&
        hgap=0, fint=0, hgapx=0.0375, fintx=0.26

bch3a: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=bangc*crtod/2, e2=0, tilt=180,&
        hgap=0.0375, fint=0.26, hgapx=0, fintx=0
bch3b: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=0, tilt=180,&
        hgap=0, fint=0, hgapx=0, fintx=0
bch3c: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=bangc*crtod/2, tilt=180,&
        hgap=0, fint=0, hgapx=0.0375, fintx=0.26

bch4a: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=bangc*crtod/2, e2=0,&
        hgap=0.0375, fint=0.26, hgapx=0, fintx=0
bch4b: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=0,&
        hgap=0, fint=0, hgapx=0, fintx=0
bch4c: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=bangc*crtod/2,&
        hgap=0, fint=0, hgapx=0.0375, fintx=0.26

bch1: line=(bch1a, 8*bch1b, bch1c)
bch2: line=(bch2a, 8*bch2b, bch2c)
bch3: line=(bch3a, 8*bch3b, bch3c)
bch4: line=(bch4a, 8*bch4b, bch4c)

```



```

-0.03
-0.02
-0.01
0
0.01
0.02
0.03
0.04
0.05
0.06
1
1,175
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0.06
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0.05
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0.04
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0.03
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0.02
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0.01
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 -0.01
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 -0.02
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 -0.03
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 -0.04
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 -0.05
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 -0.06
1 50 1
1 -1 0
1.6 1.6,
rmatrix
0 0 0 0 -0.06
1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 -0.05
1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 -0.04
1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 -0.03
1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix

```

```

0 0 0 0 0 -0.02
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 -0.01
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.0
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.01
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.02
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.03
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.04
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.05
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.06
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
hardware layout
0.2
0 0 0
0 0 0
1 0;

!25.46866438 -1.124620668 17.58194829 -0.415069892 prior to 12/08
!21.56291374 -0.766279877 21.36142827 -0.871894043 12/08 and later
!21.56291374 -0.766279877 17.69471126 -0.612901307 12/12 and later

```

stop

Appendix 2b. DIMAD input kb5_irwig2reinj.in

```
utransport
TITLE
IR Upgrade Design Revision 1.1.2, 8 October 2001

!! COMMENTS FOR REV 1.0
!! fint numbers approximately those provided by Jeff Karn
!! 32 m/16 m optical cavities for 75 MHz drive laser
!! minor interference of end-can with fat chic
!! little chic dipoles approximately DW's
!! path length 10th subharmonic/drive laser 20th subharmonic
!! so that accelerated/energy recovered bunches essentially
!! interleaved

! COMMENTS FOR REV 1.1-----
!
! injection momentum: 10 MeV/C
! driver dynamic range: 80-210 MeV/c
! baseline set point: 145 MeV/c
! note these are momenta!
!
! 1. injection line changed to sector dipoles as in IR Demo
!    per B. Yunn specifications (e-mail of 11/14/2000)
!    layout evaluated in "rev 1.1 layouts.xls"
!
! 2. linac accelerates from 10 MeV/c to 145 MeV/c
!    approximate module gains: 40 MeV, 55 MeV, 40 MeV
!
! 3. linac slots allow use of 7-cell modules in all locations
!    with 1 m (center to center) triplets in warm regions

!
! comments for rev 112
!
! 1. widened linac triplets by 2.4 cm
! 2. moved er dump chicane 0.25 m downstream
! 3. moved er line triplet 0.25 m downstream
!   (this is not in this file - it is in the dump line file)
! 4. initial quad strengths from rev111 output

pi=3.14159265358979
cdtor=pi/180
crtod=180/pi
rflambda=0.20026216299

!DRIFTS
DFAC: DRIFT, L = 8.2500

d0:drift, l=1
! adjust to set path length - delta1 in this gives 4*delta1 on pathlength:
!d0a:drift, l=0.43429490824
d0a:drift, l=0.5 + 0.0208744373 - rflambda/2 + 0.00731255025
dbcm: drift, l=(2*rflambda+0.25)

DF1: DRIFT, L=0.875 - 0.024
DF2: DRIFT, L=0.35 + 0.024
DF3: DRIFT, L=0.675 - 0.024

DA1: DRIFT, L=0.5341
DA2: DRIFT, L=0.0500
DA3: DRIFT, L=0.4606

! optics solution from "upgrade linac optics 16 November 2000 alt rev112.xls"
!21.65441457 -0.837499584 16.65441457 0.837499584 1 21.4234518 -0.956381413 17.4234518
0.956381413
KQ21 =-3.597509424
KQ31 = 6.755323252
KQ41 = 1.29434205
KQ51 =-2.550615217

KQ22 =KQ21 * 50/105.0102458
KQ32 =KQ31 * 50/105.0102458
KQ42 =KQ41 * 105/ 50.01035005
KQ52 =KQ51 * 105/ 50.01035005

Q21: QUADRUPOLE, L=lqm/6, K1=KQ21, aperture=0.0375
Q31: QUADRUPOLE, L=lqm/6, K1=KQ31, aperture=0.0375
Q41: QUADRUPOLE, L=lqm/6, K1=KQ41, aperture=0.0375
Q51: QUADRUPOLE, L=lqm/6, K1=KQ51, aperture=0.0375

Q22: QUADRUPOLE, L=lqm/6, K1=KQ22, aperture=0.0375
Q32: QUADRUPOLE, L=lqm/6, K1=KQ32, aperture=0.0375
Q42: QUADRUPOLE, L=lqm/6, K1=KQ42, aperture=0.0375
Q52: QUADRUPOLE, L=lqm/6, K1=KQ52, aperture=0.0375

!POSITION MONITORS
PMA:MONITOR,L= 0.254
PM:MONITOR,L= 0

! ACCELERATING SECTIONS
!----- 1ST PASS - UP -----
AC001:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC002:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC003:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC004:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC005:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC006:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC007:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC008:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=5.07579053182377, PHI0=-10
AC009:CEBCAV, L=0.7, EN0=ENERG0, DELTAE=6.98088346018567, PHI0=-10
```

```

AC010:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC011:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC012:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC013:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC014:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC015:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC016:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=-10
AC017:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC018:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC019:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC020:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC021:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC022:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC023:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
AC024:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=-10
!----- 2ND PASS - DOWN -----
AC025:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC026:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC027:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC028:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC029:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC030:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC031:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC032:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07579053182377, PHI0=170
AC033:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC034:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC035:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC036:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC037:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC038:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC039:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC040:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=6.98088346018567, PHI0=170
AC041:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC042:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC043:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC044:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC045:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC046:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC047:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
AC048:CEBCAV, L=0.7, ENO=ENERGO,DELTAE=5.07708968294974, PHI0=170
!LATTICE
CELLA(ACA,ACB): LINE=(ACA,DA2,ACB)
SCRYO:LINE=( DA1,CELLA(AC1,AC2),DA1)
CRYO(ACA,ACB,ACC,ACD,ACE,ACF,ACG,ACH):LINE=(DA1,&
CELLA(ACA,ACB),DA3,CELLA(ACC,ACD),DA3,&
CELLA(ACE,ACF),DA3,CELLA(ACG,ACH),DA1)

warm(q1,q2,q3): line=(df1, 6*q1, df2, 6*q2, df3, 6*q3, df4)
mraw(q1,q2,q3): line=(df3, 6*q1, df2, 6*q2, df1, 6*q3, df4)

LINAC1: LINE=(d0, &
CRYO(AC001,AC002,AC003,AC004,AC005,AC006,AC007,AC008),&
warm(q21, q31, q21), &
CRYO(AC009,AC010,AC011,AC012,AC013,AC014,AC015,AC016),&
mraw(q41, q51, q41), &
CRYO(AC017,AC018,AC019,AC020,AC021,AC022,AC023,AC024),&
d0)

LINAC2: LINE=(d0, &
CRYO(AC025,AC026,AC027,AC028,AC029,AC030,AC031,AC032),&
warm(q22, q32, q22), &
CRYO(AC033,AC034,AC035,AC036,AC037,AC038,AC039,AC040),&
mraw(q42, q52, q42), &
CRYO(AC041,AC042,AC043,AC044,AC045,AC046,AC047,AC048),&
d0)

arcfit1:marker
arcfit2:marker
!21.65441457      -0.837499584     16.65441457      0.837499584     1      21.4234518      -0.956381413     17.4234518
0.956381413
beta0x = 21.65441457
alpha0x = -0.837499584
beta0y = 21.42345187
alpha0y = -0.956381413
momcompt= 0.19205

arcmat: matrix, r11=sqrt(beta0x/beta0x), &
r12=0, &
r21=(alpha0x-alpha0x)/sqrt(beta0x*beta0x), &
r22=sqrt(beta0x/beta0x), &
r33=sqrt(beta0y/beta0y), &
r34=0, &
r43=(alpha0y-alpha0y)/sqrt(beta0y*beta0y), &
r44=sqrt(beta0y/beta0y), &
r55=momcompt, r56=0, r66=1

alllinac: line=(linac1,arcfit1,arcmat,arcfit2,linac2)

! arc data -----
injslot: drift, l=2.98963

```

```

! Excel value: injface=3.867235220330010
! dimad value:
injface=.386780036798478+01

breinj1: sbend, l=0.217416041076069, angle=1.431841557712840, &
e1=0, e2=(1.431841557712840-20), tilt=180, &
hgaps=0.0375, fint=0.3, hgapx=0.0375, fintx=0.3
breinj2: sbend, l=0.217416041076069, angle=1.431841557712840, &
e1=0, e2=injface, &
hgaps=0.0375, fint=0.3, hgapx=0, fintx=0
breinj3: sbend, l=0.217416041076069, angle=1.431841557712840, &
e1=0, e2=injface, &
hgaps=0, fint=0, hgapx=0.0375, fintx=0.3
breinj4: sbend, l=0.217416041076069, angle=1.431841557712840, &
e2=0, e1=(1.431841557712840-20), tilt=180, &
hgaps=0.0375, fint=0.3, hgapx=0.0375, fintx=0.3

dreinj1: drift, l=0.815467800213433
dreinj2: drift, l=0.5
dreinj3: drift, l=0.48963

reinj: line=(breinj1, dreinj1, breinj2, &
breinj3, dreinj1, breinj4, dreinj3)

bext1: sbend, l=0.217416041076069, angle=1.431841557712840, &
e1=0, e2=(1.431841557712840-20), &
hgaps=0.0375, fint=0.3, hgapx=0.0375, fintx=0.3
bext2: sbend, l=0.217416041076069, angle=1.431841557712840, &
e2=0, e1=injface, tilt=180, &
hgaps=0.0375, fint=0.3, hgapx=0, fintx=0
bext3: sbend, l=0.217416041076069, angle=1.431841557712840, &
e1=0, e2=injface, tilt=180, &
hgaps=0, fint=0, hgapx=0.0375, fintx=0.3
bext4: sbend, l=0.217416041076069, angle=1.431841557712840, &
e2=0, e1=(1.431841557712840-20), &
hgaps=0.0375, fint=0.3, hgapx=0.0375, fintx=0.3

extra: line=(dbcm, &
bext1, dreinj1, bext2, &
bext3, dreinj1, bext4)

lmatch1 = 6 + 1 + 0.5 - 1
lmatch2 = 17.639630 + 2 + 2*0.075904533259 + 1 - 0.2149809 &
+ 0.0631718 - 2 + 2*rflambda

lqm = 0.15

! match to arc -----
kqm11 =-.29161950608304E+01
!kqm12 = .64274479269415E+01
kqm12 = 6.5
kqm13 =-.45715119232099E+01
kqm14 = .20916472121916E+01
kqm15 =-.30425614550167E+01
kqm16 = 6

qml1: quadrupole, l=lqm/6, k1=kqm11
qml2: quadrupole, l=lqm/6, k1=kqm12
qml3: quadrupole, l=lqm/6, k1=kqm13
qml4: quadrupole, l=lqm/6, k1=kqm14
qml5: quadrupole, l=lqm/6, k1=kqm15
qml6: quadrupole, l=lqm/6, k1=kqm16

ldm11=((lmatch1/8)-(lqm/2)) + 0.2
ldm12=(0.75-lqm)
ldm13=(lmatch1/8)-lqm - 0.2

dm11alt: drift, l=(ldm11-0.25)
dm11: drift, l=ldm11
dm12: drift, l=ldm12
dm13: drift, l=ldm13

match1: line=(dm11alt, 6*qml1, dm13, 6*qml2, dm13, 6*qml3, &
dm11, qml1, &
6*qml4, dm13, 6*qml5, dm13, 6*qml6, dm11)

! match to wiggler garden-----
kqm21= -4.4 *1.125
kqm22= 4.4
kqm23= -4.4
kqm24= 4.4
kqm25= -4.4
kqm26= 4.4
kqm27= -4.4
kqm28= 4.4
kqm29= -4.4
kqm210= 4.4*0.75
kqm211=-4.4*0.5
kqm212= 4.4*0.25

qml21: quadrupole, l=lqm/6, k1=kqm21
qml22: quadrupole, l=lqm/6, k1=kqm22
qml23: quadrupole, l=lqm/6, k1=kqm23
qml24: quadrupole, l=lqm/6, k1=kqm24
qml25: quadrupole, l=lqm/6, k1=kqm25
qml26: quadrupole, l=lqm/6, k1=kqm26
qml27: quadrupole, l=lqm/6, k1=kqm27
qml28: quadrupole, l=lqm/6, k1=kqm28
qml29: quadrupole, l=lqm/6, k1=kqm29
qml210:quadrupole, l=lqm/6, k1=kqm210
qml211:quadrupole, l=lqm/6, k1=kqm211
qml212:quadrupole, l=lqm/6, k1=kqm212

ldm21=((lmatch2/12)-lqm)/2
ldm23=(lmatch2/12)-lqm

```

```

dm21: drift, l=ldm21
dm23: drift, l=ldm23

blcell: line=(3*qm26,dm23,6*qm27,dm23,3*qm26)

fm21: marker
fm22: marker
fmbc1: marker
fmbc2: marker

match2: line=(dm21, 6*qm21, dm23, &
              6*qm22, dm23, &
              6*qm23, dm23, &
              6*qm24, dm23, &
              6*qm25, dm23, &
              3*qm26, fm21, 3*qm26, dm23, &
              3*qm27, 3*qm27, dm23, &
              3*qm28, fm22, 3*qm28, dm23, &
              6*qm29, dm23, &
              6*qm210, dm23, &
              6*qm211, dm23, &
              6*qm212, dm21)

! match to ir wiggler -----
kqm31 = .29002763186331E+01
kqm32 = -.33174132988901E+01
kqm33 = 0
kqm34 = 0
kqm35 = .38024948331400E+01
kqm36 = -.34839913449125E+01

qm31: quadrupole, l=lqm/6, k1=kqm31
qm32: quadrupole, l=lqm/6, k1=kqm32
qm33: quadrupole, l=lqm/6, k1=kqm33
qm34: quadrupole, l=lqm/6, k1=kqm34
qm35: quadrupole, l=lqm/6, k1=kqm35
qm36: quadrupole, l=lqm/6, k1=kqm36

lmatch3=10

!ldm31=0.925
!ldm32=0.85
!ldm33=0.425
ldm31=(lmatch3-4)/4 - lqm/2 + 0.2
ldm31a=(lmatch3-4)/4 - lqm/2 + 0.2 + 0.14
ldm32= 1.0 - lqm - 0.2

dm31: drift, l=ldm31
dm31a: drift, l=ldm31a
dm32: drift, l=ldm32
!dm33: drift, l=ldm33

bump: drift, l=3

match3: line=(dm31, 6*qm31, dm32, 6*qm32, dm32, 6*qm33, dm31, &
              dm31, 6*qm34, dm32, 6*qm35, dm32, 6*qm36, dm31a)

! match from wiggler garden -----
!kQM51 = .50285990019186E+01
!kQM52 = -.41521663367068E+01
!kQM53 = .18066117437057E+00
!kQM54 = .49211192973563E+01
!kQM55 = -.38723341038288E+01
!kQM56 = .30319519805250E+00
!kQM51 = -2
!kQM52 = 4
!kQM53 = -2
!kQM54 = 0
!kQM55 = 3
!kQM56 = -3

kQM51 = -.12146182906751E+01
kQM52 = -.63253855991718E+01
kQM53 = .54766661899922E+01
kQM54 = -.35326532555102E+01
kQM55 = .17078204392578E+00
kQM56 = .18644842262609E+01

qm51: quadrupole, l=lqm/6, k1=kqm51
qm52: quadrupole, l=lqm/6, k1=kqm52
qm53: quadrupole, l=lqm/6, k1=kqm53
qm54: quadrupole, l=lqm/6, k1=kqm54
qm55: quadrupole, l=lqm/6, k1=kqm55
qm56: quadrupole, l=lqm/6, k1=kqm56

lmatch5=10

!ldm51=1.680637
!ldm52=1.322181
!ldm53=0.15 + 0.075
ldm51=0.5 - 0.14 + 0.24
!ldm52=1.292761222056
ldm52=1.25 - 0.15 - 0.15 - 0.1
ldm53=1.38947755588799 + (1.292761222056-1.25)*2 + 2*0.15 + 0.1 &
      - 0.12
ldm54=0.5 + 0.15 + 4*0.15 + 2*0.1

dm51: drift, l=ldm51
dm52: drift, l=ldm52
dm53: drift, l=ldm53
dm54: drift, l=ldm54

!ldm52b=0.45171286141/cos(cdtor*7.5)
!ldm52c=0.25
!dm52b: drift, l=ldm52b
!dm52c: drift, l=ldm52c

```

```

match5: line=(-(dm54, 6*qm51, dm52, 6*qm52, dm52, 6*qm53, dm53, &
dm53, 6*qm54, dm52, 6*qm55, dm52, 6*qm56, dm51))

!      bridging from optical cavity chicane to endloop

! match to reinjection -----
kQM61 = .82564998314733E+00
!kQM62 = -.45261959060859E+01
kQM62 = -5
kQM63 = -.10826098940769E+01
kQM64 = 0
kQM65 = .41236768158546E+01
kQM66 = -.38640794706814E+01

qm61: quadrupole, l=lqm/6, k1=kqm61
qm62: quadrupole, l=lqm/6, k1=kqm62
qm63: quadrupole, l=lqm/6, k1=kqm63
qm64: quadrupole, l=lqm/6, k1=kqm64
qm65: quadrupole, l=lqm/6, k1=kqm65
qm66: quadrupole, l=lqm/6, k1=kqm66

!ldm61=((lmatch1/8)-(lqm/2))
!ldm62=(0.75-lqm)
!ldm63=(lmatch1/8)-lqm

!ldm61=0.416624
!ldm62=1.179402
!ldm63=0.157883 + 0.075
ldm61=0.5
ldm62=0.9875
ldm63=0.5+0.075

dm61: drift, l=ldm61
dm62: drift, l=ldm62
dm63: drift, l=ldm63

match6: line=( 6*qm61, dm62, 6*qm62, dm62, 6*qm63, &
dm63, dm63, &
6*qm64, dm62, 6*qm65, dm62, 6*qm66, dm61)

! endloop data -----
```

rho=1
rhop=1.2
rhopp=1.2
bang=43.383565*pi/180
face1= 2
face2=16.5
face3= 0

bla: sbend, l=rhop*bang/10, angle=bang*crtod/10, &
 el=face1, e2=0, &
 hgap=0.0375, fint=0.26, hgapx=0, fintx=0
blb: sbend, l=rhop*bang/10, angle=bang*crtod/10, &
 el=0, e2=0
blc: sbend, l=rhop*bang/10, angle=bang*crtod/10, &
 el=0, e2=face1, &
 hgap=0, fint=0, hgapx=0.0375, fintx=0.26

b2a: sbend, l=rhop*bang/10, angle=bang*crtod/10, &
 el=face2, e2=0, &
 hgap=0.0375, fint=0.26, hgapx=0, fintx=0
b2b: sbend, l=rhop*bang/10, angle=bang*crtod/10, &
 el=0, e2=0, &
 tilt=180
b2c: sbend, l=rhop*bang/10, angle=bang*crtod/10, &
 el=0, e2=face2, &
 tilt=180, &
 hgap=0, fint=0, hgapx=0.0375, fintx=0.26

b1: line=(bla, 8*b1b, blc)
b2: line=(b2a, 8*b2b, b2c)

bb1: sbend, l=rho*pi/60, angle=pi*crtod/60, el= face3, e2=0,&
 hgap=0.0375, fint=0.26, hgapx=0, fintx=0
bb2: sbend, l=rho*pi/60, angle=pi*crtod/60, el=0, e2=0
bb3: sbend, l=rho*pi/60, angle=pi*crtod/60, el=0, e2=0
bb4: sbend, l=rho*pi/60, angle=pi*crtod/60, el=0, e2=0
bb5: sbend, l=rho*pi/60, angle=pi*crtod/60, el=0, e2= face3,&
 hgap=0, fint=0, hgapx=0.0375, fintx=0.26

ldell1 =0.28514265396274E+00
ldella=0.25
ldellb=0.075
ldel12 =0.29846876655049E+00
ldel2a=0.25
ldel2b=0.075

dell1: drift, l=ldell1
della: drift, l=ldella
dellb: drift, l=ldellb
del2: drift, l=ldel2
del2a: drift, l=del2a
del2b: drift, l=del2b

kqt11 = .11887968731297E-01
kqt12 =-.80401660588917E-01
kqt21 =-.20510638424916E-02
kqt22 = .13813705765973E-01

qt11a: quadrupole, l=0.15/6, k1=kqt11
qt11b: quadrupole, l=0.15/6, k1=kqt11
qt12a: quadrupole, l=0.15/6, k1=kqt12
qt12b: quadrupole, l=0.15/6, k1=kqt12
qt21a: quadrupole, l=0.15/6, k1=kqt21
qt21b: quadrupole, l=0.15/6, k1=kqt21
qt22a: quadrupole, l=0.15/6, k1=kqt22
qt22b: quadrupole, l=0.15/6, k1=kqt22

```

ks11a =-.66817708471406E+01
ks12a = .10648527879621E+02
ks11b =.11431367987357E+02
ks11b =-.87268253610551E+01
ks21a =-.66776830601453E+01
ks22a = .92813942252631E+01
ks22b = .92871457140399E+01
ks21b =-.66934000325820E+01

s11a: sextupole, l=0.15/6, k2=ks11a
s11b: sextupole, l=0.15/6, k2=ks11b
s12a: sextupole, l=0.15/6, k2=ks12a
s12b: sextupole, l=0.15/6, k2=ks12b
s21a: sextupole, l=0.15/6, k2=ks21a
s21b: sextupole, l=0.15/6, k2=ks21b
s22a: sextupole, l=0.15/6, k2=ks22a
s22b: sextupole, l=0.15/6, k2=ks22b

fitbb: marker

endloop1: line=(d0a, b1, dell, (2*dellb), della, (6*s11a, della, 6*qt11a), &
               dell, b2, &
               del2, 2*del2b, del2a, 6*s12a, del2a, 6*qt12a, del2, &
               bbl, 14*bb2, 15*bb3, fitbb, 15*bb3, 14*bb4, bb5,&
               del2, 6*qt12b, del2a, 6*s12b, del2a, 2*del2b, del2, &
               b2, dell, (6*qt11b,della, 6*s11b), della, (2*dellb), dell, b1, d0a)

oo: multipole, l=0, k3=-1.7
!oo: multipole, l=0, k3=0

endloop2: line=(d0a, b1, dell, (2*dellb), della, (6*s21a, della, 6*qt21a), &
               dell, b2, &
               del2, del2b, oo, del2b, del2a, 6*s22a, del2a, 6*qt22a, del2, &
               bbl, 14*bb2, 15*bb3, fitbb, 15*bb3, 14*bb4, bb5,&
               del2, 6*qt22b, del2a, 6*s22b, del2a, del2b, oo, del2b, del2, &
               b2, dell, (6*qt21b, della, 6*s21b), della, (2*dellb), dell, b1, d0a)

fit1:marker
fit2:marker
fit3:marker
fit4:marker
fit5:marker
fit6:marker
fit7:marker

! optical cavity chicane

bangx=23.13749962
facex=-18.24606538
bangc=20.24606538*cotor
!= (43.whatever-bangx)*cotor

rhoc=1.2

ldch1=0
ldch2=1.779414643-ldch1/cos(bangc)
ldch3=0.5

dch1: drift, l=ldch1
dch2: drift, l=ldch2
dch3: drift, l=ldch3

bch1a: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=bangc*crtod/2, e2=0, &
        hgap=0.0375, fint=0.26, hgpx=0, fintx=0
bch1b: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=0,&
        hgap=0, fint=0, hgpx=0, fintx=0
bch1c: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=bangc*crtod/2,&
        hgap=0, fint=0, hgpx=0.0375, fintx=0.26
bch2a: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=bangc*crtod/2, e2=0, tilt=180,&
        hgap=0.0375, fint=0.26, hgpx=0, fintx=0
bch2b: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=0, tilt=180,&
        hgap=0, fint=0, hgpx=0, fintx=0
bch2c: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=bangc*crtod/2, tilt=180,&
        hgap=0, fint=0, hgpx=0.0375, fintx=0.26
bch3a: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=bangc*crtod/2, e2=0, tilt=180,&
        hgap=0.0375, fint=0.26, hgpx=0, fintx=0
bch3b: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=0, tilt=180,&
        hgap=0, fint=0, hgpx=0, fintx=0
bch3c: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=bangc*crtod/2, tilt=180,&
        hgap=0, fint=0, hgpx=0.0375, fintx=0.26
bch4a: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=bangc*crtod/2, e2=0,&
        hgap=0.0375, fint=0.26, hgpx=0, fintx=0
bch4b: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=0,&
        hgap=0, fint=0, hgpx=0, fintx=0
bch4c: sbend, l=rhoc*bangc/10, angle=bangc*crtod/10, &
        el=0, e2=bangc*crtod/2,&
        hgap=0, fint=0, hgpx=0.0375, fintx=0.26

bch1: line=(bch1a, 8*bch1b, bch1c)
bch2: line=(bch2a, 8*bch2b, bch2c)
bch3: line=(bch3a, 8*bch3b, bch3c)
bch4: line=(bch4a, 8*bch4b, bch4c)

ch2: line=(dch1, bch1, dch2, bch2, dch3, &
           fmbo1, dch3, bch3, dch2, bch4, dch1, fmbo2)

start:marker

```

```

!dwig: drift, l=6/2
wigdrifl: drift, l=0.6
wigdrif2: drift, l=0.6
rhowig=2.78163968113
bangwig=1.02994765529

wig1: sbend, l=rhowig*bangwig*cdtor, angle=bangwig, &
      el=0, e2=bangwig, tilt=90
wig2: sbend, l=2*rhowig*bangwig*cdtor, angle=2*bangwig, &
      el=bangwig, e2=bangwig, tilt=-90
wig3: sbend, l=rhowig*bangwig*cdtor, angle=bangwig, &
      el=0, e2=bangwig, tilt=90

wigper: line=(wig1,wig2,wig3)

dwig: line=(wigdrifl, 12*wigper)
giwd: line=(12*wigper, wigdrif2)

ltowig: line=(start,extra, match1, fit1, endloop1, fit2, match2, &
              ch2, match3, dwig)
rc:line=(giwd, match5, &
         fit1, endloop2, fit2, &
         match6, reinj)

coll1:ecollimator, l=0,xsize=0.0254, ysize=0.0254
coll2:ecollimator, l=0,xsize=0.0254*0.80,ysize=0.0254*0.80
coll3:ecollimator, l=0,xsize=0.0254*0.70,ysize=0.0254*0.70
coll4:ecollimator, l=0,xsize=0.0254*0.60,ysize=0.0254*0.60
coll5:ecollimator, l=0,xsize=0.0254*0.50,ysize=0.0254*0.50

rclinac: line=(rc,linac2, coll1, coll2, coll3, coll4, coll5)

recirc: line=(ltowig, rc)
origin:marker
determ=.1011280E+01*.7795121E-01-.5197606E+01*.1897985E-02

fixit: matrix, r11=1,r22=1,r33=1,r44=1,&
r55=-.7795121E-01/determ,r66=.1011280E+01/determ,&
r56=-.1897985E-02/determ,r65=-.5197606E+01/determ

all: line=(origin, linac1, fixit, recirc, linac2)

!kbb.....!!!!KBB; bends up to, but not including, IR wiggler proper
!bend2irwig: line=(start,extra, match1, fit1, endloop1, &
!                    fit2, match2, ch2, match3, wigdrif1, irmark1)
bend2irwig: line=(start,extra, match1, fit1, endloop1, fit2, match2, &
                   ch2, match3, wigdrif1, irmark1)

!      1   2   3   4   5   6   7   8
!234567890123456789012345678901234567890123456789012345678901234567890
!bend2irwig: line=(d0a, b1, dell1, (2*dell1b), delta1, (6*s11a, delta1, 6*qt11a), &
!                   dell1, b2, dell2, 2*dell2b, del2a, 6*s12a, del2a, 6*qt12a, del2,&
!                   bb1, 14*bb2, 15*bb3, fitbb, 15*bb3, 14*bb4, bb5,&
!                   dell2, 6*qt12b, del2a, 6*s12b, del2a, 2*del12b, del2, b2, dell1,&
!                   (6*qt11b,delta1, 6*s11b), delta1, (2*dell1b), dell1, b1, d0a)
bend2irwig: line=(start,extra, match1, fit1, endloop1, fit2, match2, &
                   ch2, match3, wigdrif1)

!!!!KBB IR wiggler proper
!
irwig: line=(wig1,wig2,wig3)

!!!!KBB; after IR wiggler proper, through bends, to before reInjection
!
irwig2reinj: line=(wigdrif2, match5, fit1, endloop2, fit2, match6)
!
!kbb.....use, irwig2reinj

dimat
matrix
2 -1,
machine
1 2 1 0 1 1 1
1.724 -1.1678 0 0
1.744 .0000 0 0
0,
beam definition - 30 mm mrad normalised at 150 MeV
0
1.724 -1.1678 0 0 0.1e-6
1.744 .0000 0 0 0.1e-6
0.00012 0.02
1000,
detailed analysis
1 1 1
0 0 0 0
1.0e-06 1.0e-06 1.0e-06 1.0e-06
1.724 -1.1678 0 0
1.744 .0000 0 0
13 0
-0.06
-0.05
-0.04
-0.03
-0.02
-0.01
0
0.01

```

```

0.02
0.03
0.04
0.05
0.06
1,
1,175
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 0.06
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 0.05
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 0.04
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 0.03
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 0.02
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 0.01
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 0
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 -0.01
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 -0.02
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 -0.03
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 -0.04
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 -0.05
1 50 1
1 -1 0
1.6 1.6,
line geometric aberrations
1.724 -1.1678 1.744 .0000
0 0 0 0 -0.06
1 50 1
1 -1 0
1.6 1.6,
rmatrix
0 0 0 0 0 -0.06
1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 -0.05
1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 -0.04
1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 -0.03
1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 -0.02
1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix

```

```

0 0 0 0 0 -0.01
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.01
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.02
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.03
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.04
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.05
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
rmatrix
0 0 0 0 0 0.06
1e-5 1e-5 1e-5 1e-5 1e-5 1e-5
1 -1
0,
hardware layout
0.2
0 0 0 0
0 0 0
1 0;

!25.46866438 -1.124620668 17.58194829 -0.415069892 prior to 12/08
!21.56291374 -0.766279877 21.36142827 -0.871894043 12/08 and later
!21.56291374 -0.766279877 17.69471126 -0.612901307 12/12 and later

```

stop

Appendix 3. Source Code Description

The development tree for tdbbu is organized in this author's standard method¹⁷. As usual, the source is kept as a collection of files with one routine in each file. All shared FORTRAN PARAMETERS, COMMONs, and their declarations are kept in INCLUDE files. A Makefile is used to recompile the code and update the documentation. The author's KWRAP¹⁸ and KBB¹⁹ libraries provide a command line interface and platform independence respectively.

The author's splitcf²⁰ program is used to generate some of the documentation and to check the COMMON blocks. It analyzes the source code and outputs a web page with hyperlinks to each of the files; a copy follows.

```
splitcf +mc 2 -C+ tdbbuk_main.c SRC/fc_ranf.c SRC/fc_ranset.c -F+ SRC/arraycheck.f SRC/bunch_info.f SRC/cleanlist.f SRC/clockk.f SRC/erase_all_bunches.f  
SRC/erase_logged.f SRC/estimate_t_nextstep.f SRC/gaussian1d.f SRC/generate_bunch.f SRC/guessnext.f SRC/guessreport.f SRC/guessshow.f SRC/hit_wall.f  
SRC/hom_force_update.f SRC/hom_operate.f SRC/hom_reset_all.f SRC/hom_summary.f SRC/keep_killedstats.f SRC/m4x4_print.f SRC/print_path.f SRC/precalc.f  
SRC/print_path.f SRC/propagate_bunch.f SRC/ranf.f SRC/ranset.f SRC/report_recirculationtimes.f SRC/rms2slope.f SRC/rough_estimate.f SRC/rtm_operate.f SRC/rtm_print.f  
SRC/second.f SRC/skipoverlines.f SRC/sortincreasing.f SRC/stepk.f SRC/step_record.f SRC/step_report.f SRC/tdbbu_dumpbunch.f SRC/tdbbu_ext_rim.f SRC/tdbbu_ext_tm.f  
SRC/tdbbu.f SRC/tdbbuk.f SRC/tdbbu_logbunch.f SRC/tdbbu_recall_pz.f SRC/tdbbu_rescan_plots.f SRC/tdbbu_scanmatrixfile.f SRC/timeorder.f SRC/tm_operate.f  
SRC/verify_space.f SRC/whereat.f SRC/whereindexmap.f SRC/wipe_all.f SRC/yesno.f +in SRC +ls +j +k +v +W SRC_3.html
```

Wed Oct 13 13:51:00 2004

[\[tdbbuk_main\]](#)

[\[descriptions of routines\]](#)

tdbbuk_main

<u>main</u>	<u>TDBBUK</u>	<u>YESNO</u>							
		<u>RANSET</u>							
		<u>CLEANLIST</u>							
		<u>TDBBU RESCA N PLOTS</u>	<u>RMS2SLOPE</u>						
		<u>TDBBU</u>	<u>YESNO</u>						
			<u>SECOND</u>						
			<u>WIPE_ALL</u>	<u>HOM_RESET_ALL</u>					
				<u>ERASE_ALL_BUNCHES</u>					
				<u>ERASE_LOGGED</u>					
				<u>KEEP_KILLEDSTATS</u>					

17 JLAB-TN-03-001, *My Code Development Style, Organization, and Platform Dependency Guide*, K.B. Beard

18 Jlab CUE: /u/group/casa/SUPPORTED/KWRAP/index.html

19 Jlab CUE: /u/group/casa/SUPPORTED/KBB/index.html

20 Jlab CUE: /u/group/casa/SUPPORTED/miscellaneous/index.html

	<u>CLOCKK</u>				
	<u>PRECAL</u>	<u>YESNO</u>			
		<u>RTM PRINT</u>	<u>M4X4 PRINT</u>		
		<u>ARRAYCHECK</u>	<u>WHEREAT</u>	<u>WHEREINDEXMAP</u>	
	<u>PLAN PATH</u>	<u>YESNO</u>			
		<u>PRINT PATH</u>			
	<u>TDBBU SCANMATTR</u>	<u>YESNO</u>			
	<u>XFILE</u>				
		<u>SKIPOVERLINES</u>			
		<u>SORTINCREASING</u>			
		<u>TDBBU RECALL_PZ</u>			
		<u>M4X4 PRINT</u>			
		<u>TDBBU EXT RTM</u>	<u>M4X4 PRINT</u>		
	<u>ROUGH ESTIMATE</u>				
	<u>VERIFY SPACE</u>				
	<u>STEPK</u>	<u>YESNO</u>			
		<u>KEEP_KILLEDSTATS</u>			
		<u>PLAN PATH</u>	<u>YESNO</u>		
			<u>PRINT PATH</u>		
		<u>GENERATE_BUNCH</u>	<u>RANE</u>		
		<u>ESTIMATE_T_NEXSTEP</u>	<u>TM OPERATE</u>	<u>YESNO</u>	
		<u>BUNCH_INFO</u>			
		<u>PROPAGATE_BUNCH</u>	<u>BUNCH_INFO</u>		
			<u>HOM OPERATE</u>	<u>YESNO</u>	
				<u>TIMEORDER</u>	<u>YESNO</u>
			<u>TDBBU EXT RTM</u>	<u>M4X4 PRINT</u>	
			<u>RTM OPERATE</u>	<u>M4X4 PRINT</u>	
			<u>TDBBU EXT TM</u>	<u>RANF</u>	
				<u>GAUSSIANID</u>	<u>RANF</u>
			<u>TM OPERATE</u>	<u>YESNO</u>	
			<u>ESTIMATE_T_NEXSTEP</u>	<u>TM OPERATE</u>	<u>YESNO</u>
			<u>TDBBU LOGBUNCH</u>	<u>SORTINCREASING</u>	
		<u>TDBBU DUMPBUNCH</u>			
		<u>HOM OPERATE</u>	<u>YESNO</u>		
			<u>TIMEORDER</u>	<u>YESNO</u>	
		<u>HIT WALL</u>	<u>TDBBU_SORTDEATH_STATS</u>		
		<u>STEP RECORD</u>	<u>YESNO</u>		
			<u>TDBBU LOGBUNCH</u>	<u>SORTINCREASING</u>	
		<u>HOM_FORCE_UPDATE</u>	<u>HOM OPERATE</u>	<u>YESNO</u>	
				<u>TIMEORDER</u>	<u>YESNO</u>
		<u>STEP REPORT</u>	<u>STEP RECORD</u>	<u>YESNO</u>	
				<u>TDBBU LOGBUNCH</u>	<u>SORTINCREASING</u>
		<u>REPORT_RECIRCULATIONTIMES</u>			
	<u>HOM_SUMMARY</u>				
	<u>RMS2SLOPE</u>				
	<u>PRINT_PATH</u>				
	<u>VERIFY_SPACE</u>				
	<u>GUESSNEXT</u>	<u>TDBBU_CHECKLOSSSES</u>			
		<u>GUESSREPORT</u>			
	<u>GUESSHOW</u>				

Key: [c_routine] [FORTRAN_routine] [recursive] ...linking: FORTRAN: XXXX(...) => C: xxxx_(...)

[[SRC/arrauchek.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_io.par](#)]

SUBROUTINE ARRAYCHECK(*LABEL*, *ARRAY*, *SIZE*, *SUM*, *PRODUCT*)

CHARACTER* (*) *LABEL* - !(input) debugging label for array

REAL*8 *ARRAY(*)* - !(input) any size array

INTEGER*8 *SIZE* - !(input) number of elements within array

REAL*8 *SUM* - !(output) sum of all elements

REAL*8 *PRODUCT* - !(output) product of all elements

*
* A simple routine to look for corruption in an
* array; returns the sum of all the elements and
* the product of all the elements
*

called by: [PRECAL](#)

calls: [WHEREAT](#)

commons: [MATRIX UNITS](#)

[[SRC/bunch_info.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_bunch.cmn](#)] [[tdbbu_path.cmn](#)] [[tdbbu_time.cmn](#)]

SUBROUTINE BUNCH_INFO(*ID*, *TA*, *TX*, *DESC*, *LDESC*)

INTEGER*4 *ID* - !(input) bunch ID#

REAL*8 *TA* - !(input) current time of bunch [uS]

REAL*8 *TX* - !(input) projected time after next step [uS]

CHARACTER* (*) *DESC* - !(output) description (should be >132 characters)

INTEGER*4 *LDESC* - !(output) significant length of description

*
* Given the id# of a bunch, return a brief description
* of length Ldesc
*

called by: [PROPAGATE BUNCH](#) , [STEPK](#)

commons: [ABOUT BUNCHES](#), [ALL BUNCHES](#), [PATH](#), [MISC PATH](#), [TIME INFO](#)

[[SRC/cleanlist.f](#)]

SUBROUTINE CLEANLIST(*NV*, *V8*, *FRMT*, *DEL*, *LINE*, *LLINE*)

```

INTEGER*4 NV - !(input) # of values
REAL*8 V8(*) - !(input) values to write
CHARACTER* (*) FRMT - !(input) FORMAT -
'(f25.15)'|(e15.5)'|blank->f25.5
CHARACTER* (*) DEL - !(input) delimiter - ','|'|'...
CHARACTER* (*) LINE - !(output) string - '1.,2.,88.03'
INTEGER LLINE - !(output) important length of string

*
* Given a list of numbers, write them out using format frmt
* for each one, remove trailing zeros, pack together with
* delimiter del, and return the string line.
*
*
```

called by: [TDBBUK](#)

[[SRC/clockk.f](#)]

SUBROUTINE **CLOCKK(STRING)**

```
CHARACTER* (*) STRING - !(output) hh:mm:ss
```

```

*
* Returns the time as "hh:mm:ss" ("18:12:55" for example)
* A kludge for the CLOCK system call, uses the KBB library.
* KBB 7/22/02, modified 8/18/03
*
```

called by: [TDBBU](#)

[[SRC/erase_all_bunches.f](#)]

[[tdbbu_defs.par](#) | [tdbbu_generic.par](#) | [tdbbu_bunch.par](#) | [tdbbu_bunch.cmn](#) | [tdbbu_io.par](#)]

SUBROUTINE **ERASE_ALL_BUNCHES()**

```

*
* Zero out all bunch info.
*
```

called by: [WIPE_ALL](#)

commons: [ABOUT BUNCHES](#), [ALL BUNCHES](#), [MATRIX UNITS](#)

[[SRC/erase_logged.f](#)]

[[tdbbu_defs.par](#) | [tdbbu_generic.par](#) | [tdbbu_bunch.par](#) | [tdbbu_logged.cmn](#) | [tdbbu_io.par](#)]

SUBROUTINE **ERASE_LOGGED()**

```

*
* Zero out all bunch info.
*
```

called by: [WIPE_ALL](#)

commons: [LOGGED](#), [LOGGED LOAD](#), [MATRIX UNITS](#)

[[SRC/estimate_t_nextstep.f](#)
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_io.par](#)] [[tdbbu_1bu8.par](#)] [[tdbbu_path.cmn](#)] [[tdbbu_tm.par](#)] [[tdbbu_tm.cmn](#)]
[[tdbbu_time.cmn](#)] [[tdbbu_delays.cmn](#)]

SUBROUTINE ESTIMATE_T_NEXTSTEP(*BNCH*, *STEP*, *TNEXT*)

REAL*8 *BNCH(BEGIN:END)* - !(input) bunch
INTEGER*4 *STEP* - !(input) current step#
REAL*8 *TNEXT* - !(output) time after the next step [uS]

*
* Given a bunch *bnch* at *step#step*, estimate the time
* after *step#step*
*

called by: [PROPAGATE BUNCH](#) , [STEPK](#)

calls: [TM OPERATE](#)

commons: [MATRIX UNITS](#), [PATH](#), [MISC PATH](#), [T M INFO](#), [T M](#),
[TIME INFO](#), [DELAYS](#)

[[SRC/gaussian1d.f](#)
[[tdbbu_defs.par](#)]

REAL*8 FUNCTION GAUSSIAN1D(*SIGMA*, *CUTOFF*)

REAL*8 *SIGMA* - !(input) sigma of distribution
REAL*8 *CUTOFF* - !(input) +/- cutoff of distribution

*
* Returns a number distributed according to
* a gaussian of mean=0 and width sigma. If cutoff>0,
* distribution truncated at +/- cutoff by trying again.
* [uses Box Muller technique]
* (Borrowed from Parmela's RANNOR routine) KBB 6/21/04
*

called by: [TDBBU EXT TM](#)

calls: [RANF](#)

[[SRC/generate_bunch.f](#)
[[tdbbu_generic.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_defs.par](#)] [[tdbbu.cmn](#)] [[tdbbu_io.par](#)]

SUBROUTINE GENERATE_BUNCH(*TIME*, *QBNCH*, *BNCH*)

REAL*8 *TIME* - !(input) create time [uS]
REAL*8 *QBNCH* - !(input) fractional charge in bunch (0-1)
REAL*8 *BNCH(BEGIN:END)* - !(output) bunch

*
* Generate an initial bunch

*

called by: [STEPK](#)

calls: [RANF](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#), [MATRIX UNITS](#)

[[SRC/quessnext.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_guess.par](#)] [[tdbbu_cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_hunt.cmn](#)] [[tdbbu_enough.cmn](#)] [
[tdbbu_cuts.cmn](#)] [[tdbbu_axes.cmn](#)] [[tdbbu_duty.cmn](#)] [[tdbbu_opts.par](#)] [[tdbbu_opts.cmn](#)]]

SUBROUTINE GUESSNEXT(*IO*, *JO*, *NXPTS*, *XSLP*, *XEXP*, *NYPTS*,
YSLP, *YEXP*, *JNEXT*, *KEEPON*, *THRX*, *THRY*, *CMTX*, *CMTY*)

INTEGER *IO* - !(input) I/O channel for messages (if>0)
REAL*8 *JO* - !(input) recent current [mA]
INTEGER*8 *NXPTS* - !(input) number of X points
REAL*8 *XSLP* - !(input) recent Xrms slope result
REAL*8 *XEXP* - !(input) recent ln(Xrms) slope result
INTEGER*8 *NYPTS* - !(input) number of Y points
REAL*8 *YSLP* - !(input) recent Yrms slope result
REAL*8 *YEXP* - !(input) recent ln(Yrms) slope result
REAL*8 *JNEXT* - !(output) next current [mA]
LOGICAL *KEEPON* - !(output) whether to continue or quit
REAL*8 *THRX(LO:HI)* - !(output) best X threshold [mA]
REAL*8 *THRY(LO:HI)* - !(output) best Y threshold [mA]
CHARACTER*(*) *CMTX* - !(output) comment on X axis
CHARACTER*(*) *CMTY* - !(output) comment on Y axis

*
* A simple routine to choose whether to continue and
* suggest what current to try next.
*
* Generally, the current space ought be of the form:
*
* -----J----->
*
* [--STABLE--] [--CANT_SAY--] [--UNSTABLE--] (w/o laser)
* or [--STABLE-----] [-----UNSTABLE--] (w/o laser)
* or [--CANT_SAY-----] [-----UNSTABLE--] (w/ laser)
*
* and this algorithm turns the current up or down
* looking for the best STABLE/UNSTABLE limits, unless
* HuntForBlowup is set - in which case it looks for
* CANT_SAY/UNSTABLE limit.
*
* The CloseEnoughFraction determines the size of the
* CANT_SAY region, but the results are quite insensitive
* to its choice. When lasing, the RMS slope values gets "noisy",
* and there appears no STABLE region.
*

called by: [TDBBUK](#)

calls: [TDBBU CHECKLOSSSES](#) , [GUESSREPORT](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#),
[MATRIX UNITS](#), [HISTORY](#), [DEADZONE](#), [CUTS](#), [ONLYCONSIDER](#),
[DUTY](#), [HOM](#), [IFHITWALL](#)

[[SRC/quessreport.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_guess.par](#)] [[tdbbu_io.par](#)] [[tdbbu_hunt.cmn](#)]
SUBROUTINE GUESSREPORT(*JO*, *AXIS*, *DED*, *BST*, *REPORT*)

REAL*8 *JO* - !(input) current J [mA]
INTEGER*4 *AXIS* - !(input) either X or Y axis
INTEGER*4 *DED*(*LO*:*HI*,*X*:*Y*) - !(input) current dead zone
INTEGER*4 *BST*(*LO*:*HI*,*X*:*Y*) - !(input) current best limits
CHARACTER*(*) *REPORT* - !(output) line showing the regions

*
* Writes the current status of the dead and
* best regions into "report" for an axis (X or Y)
*
*ex: "X stable:(1)1.000 dead:(2)2.000-(3)2.500 unstable:(4)3.000"
*

called by: [GUESSNEXT](#)

commons: [MATRIX UNITS](#), [HISTORY](#)

[[SRC/quessshow.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_guess.par](#)] [[tdbbu_cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_hunt.cmn](#)] [[tdbbu_duty.cmn](#)]
SUBROUTINE GUESSSHOW(*PREFIX*, *O*, *SUFFIX*)

CHARACTER*(*) *PREFIX* - !(input) optional prefix to precede each
line
INTEGER *O* - !(input) already opened channel
CHARACTER*(*) *SUFFIX* - !(input) optional suffix to follow each line

*
* Just write the guess status to IO
*

called by: [TDBBUK](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#),
[MATRIX UNITS](#), [HISTORY](#), [DUTY](#), [HOM](#)

[[SRC/hit_wall.f](#)]
[[tdbbu_generic.par](#)] [[tdbbu_defs.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_bunch.cmn](#)] [[tdbbu_blowup.cmn](#)] [[tdbbu_path.cmn](#)] [[tdbbu_io.par](#)] [
[tdbbu_opts.cmn](#)]

SUBROUTINE HIT_WALL(*STILLINSIDE*, *MSG*)

LOGICAL*4 *STILLINSIDE* - !(output) whether the beam has hit the wall

CHARACTER* (*) *MSG* - !(output) message about the hit

```
*  
*   Return whether the beam has exceeded 100x the  
*   aperature...  
*
```

called by: [STEPK](#)

calls: [TDBBU_SORTDEATHSTATS](#)

commons: [ABOUT BUNCHES](#), [ALL BUNCHES](#), [BLOWUP](#), [PATH](#), [MISC PATH](#), [MATRIX UNITS](#), [IFHITWALL](#)

[[SRC/hit_wall.f](#)]
[[tdbbu_generic.par](#)] [[tdbbu_defs.par](#)] [[tdbbu_path.cmn](#)]

SUBROUTINE TDBBU_SORTDEATHSTATS(*LISTING*)

CHARACTER* (*) *LISTING* - !output list of deaths step(#victums),...

```
*  
*   Return a string listing the steps where deaths occurred,  
*   starting with the largest.  
*
```

called by: [HIT WALL](#)

commons: [PATH](#), [MISC PATH](#)

[[SRC/hit_wall.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_bunch.cmn](#)] [[tdbbu_opts.cmn](#)]

SUBROUTINE TDBBU_CHECKLOSSES(*EXCESSIVE*) whether beam losses exceeded tolerances

```
*  
*   Just returns whether beam losses exceeded the  
*   specified limit.  
*
```

called by: [GUESSNEXT](#)

commons: [ABOUT BUNCHES](#), [ALL BUNCHES](#), [IFHITWALL](#)

[[SRC/hom_force_update.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_cf.par](#)] [[tdbbu_cf.cmn](#)] [[tdbbu_io.par](#)]

SUBROUTINE HOM_FORCE_UPDATE(*TNOW*)

REAL*8 **TNOW** - !(input) current absolute time [uS]

```
*  
* Update all HOMs to at least Tnow, even if nothing has gone by.  
*
```

called by: [STEPK](#)

calls: [HOM OPERATE](#)

commons: [HOM MAP](#), [HOM INFO](#), [HOM WHEN](#), [HOM KICKS](#),
[WENT BY HOM](#), [MATRIX UNITS](#)

```
[ SRC/hom_operate.f ]  
[ tdbbu_generic.par ] [ tdbbu_bunch.par ] [ tdbbu_bunch.cmn ] [ tdbbu_cf.par ] [ tdbbu_cf.cmn ] [ tdbbu_tm.par ] [ tdbbu_tm.cmn ] [ tdbbu_defs.par ]  
[ tdbbu_bug.cmn ] [ tdbbu_io.par ] [ tdbbu_path.cmn ] [ tdbbu_time.cmn ] [ tdbbu_opts.par ]
```

SUBROUTINE HOM_OPERATE(**HOMID**, **ID**, **ACT**)

INTEGER*4 **HOMID** - !(input) element id#

INTEGER*4 **ID** - !(input) bunch id#

LOGICAL*4 **ACT** - !(input) store info, or act on info

```
*  
* Store the list of bunches that passed by this HOM;  
* and only Act to update the HOM and kick the bunches  
* when requested.  
* 6/04 KBB  
*
```

called by: [HOM FORCE UPDATE](#) , [PROPAGATE BUNCH](#) , [STEPK](#)

calls: [YESNO](#) , [TIMEORDER](#)

commons: [ABOUT BUNCHES](#), [ALL BUNCHES](#), [HOM MAP](#),
[HOM INFO](#), [HOM WHEN](#), [HOM KICKS](#), [WENT BY HOM](#),
[T M INFO](#), [T M](#), [BUGGY](#), [MATRIX UNITS](#), [PATH](#), [MISC PATH](#),
[TIME INFO](#)

```
[ SRC/hom_reset_all.f ]  
[ tdbbu_defs.par ] [ tdbbu_generic.par ] [ tdbbu_bunch.par ] [ tdbbu_cf.par ] [ tdbbu_cf.cmn ] [ tdbbu_io.par ]
```

SUBROUTINE HOM_RESET_ALL()

```
*  
* Reset all HOMs to an initial, empty state.  
*
```

called by: [WIPE ALL](#)

commons: [HOM MAP](#), [HOM INFO](#), [HOM WHEN](#), [HOM KICKS](#),
[WENT BY HOM](#), [MATRIX UNITS](#)

[[SRC/hom_summary.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_cf.par](#)] [[tdbbu_cf.cmn](#)] [[tdbbu_io.par](#)]

SUBROUTINE HOM_SUMMARY(*IO*)

INTEGER *IO* - !(input) already prepared I/O channel, if any

```
*  
*   Reset all HOMs to an initial, empty state.  
*
```

called by: [TDBBU](#)

commons: [HOM MAP](#), [HOM INFO](#), [HOM WHEN](#), [HOM KICKS](#),
[WENT BY HOM](#), [MATRIX UNITS](#)

[[SRC/keep_killedstats.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu.cmn](#)] [[tdbbu_bunch.par](#)] [[tdbbu_bunch.cmn](#)] [[tdbbu_bunch_desc.cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_killedstats.cmn](#)]

SUBROUTINE KEEP_KILLEDSTATS(*CLK*, *ID*, *IO*)

INTEGER*8 *CLK* - !(input) master clock step

INTEGER*4 *ID* - !(input) bunch ID#

INTEGER *IO* - !(input) already opened I/O channel, if any

```
*  
*   Keep recirculation time statistics of bunch#id if id  
*   valid; if no IO and id invalid, reset, otherwise, just  
*   print current results.  
*
```

called by: [STEPK](#) , [WIPE ALL](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#),
[ABOUT BUNCHES](#), [ALL BUNCHES](#), [MATRIX UNITS](#),
[TDBBU KILLEDSTATS](#), [TDBBU LIFETIME](#), [RECIRC STATS](#)

[[SRC/m4x4_print.f](#)]
[[tdbbu_1by16.par](#)]

SUBROUTINE M4X4_PRINT(*IO*, *COMMENT*, *M4X4*) prepared I/O
channel comment to append matrix

```
*  
*   Print a 1x16 array as 4x4 to channel IO  
*
```

called by: [RTM OPERATE](#) , [RTM PRINT](#) , [TDBBU EXT RTM](#) ,
[TDBBU SCANMATRIXFILE](#)

[[SRC/plan_path.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_beaminfo.cmn](#)] [[tdbbu.cmn](#)] [[tdbbu_time.cmn](#)] [[tdbbu_tm.par](#)] [[tdbbu_tm.cmn](#)] [[tdbbu_bug.cmn](#)]
[[tdbbu_io.par](#)] [[tdbbu_path.cmn](#)]

SUBROUTINE PLAN_PATH()

```

*
* Build the path for the bunches to follow;
* store the information required for each step.
*
* 6/04 KBB
*

```

called by: [STEPK](#) , [TDBBU](#)

calls: [YESNO](#) , [PRINT_PATH](#)

commons: [BEAM INFO](#), [GENERAL INPUT INFO](#), [DLOAD](#), [R](#),
[GENERAL INFO](#), [TIME INFO](#), [T M INFO](#), [T M](#), [BUGGY](#), [MATRIX UNITS](#),
[PATH](#), [MISC PATH](#)

```

[ SRC/precalf
  [ tdbbu\_generic.par ] [ tdbbu\_cmn ] [ tdbbu\_time.cmn ] [ tdbbu\_beaminfo.cmn ] [ tdbbu\_tm.par ] [ tdbbu\_tm.cmn ] [ tdbbu\_delays.cmn ] [
  tdbbu\_rtm.cmn ] [ tdbbu\_bug.cmn ] [ tdbbu\_defs.par ] [ tdbbu\_io.par ] [ tdbbu\_est.cmn ] [ tdbbu\_duty.cmn ] [ tdbbu\_1by16.par ] [ tdbbu\_lasing.cmn ] [
  tdbbu\_bunch.par ] [ tdbbu\_cf.par ] [ tdbbu\_cf.cmn ] [ tdbbu\_print.cmn ] [ tdbbu\_matrix.cmn ]

```

SUBROUTINE PRECAL([IN](#) , [O](#) , [NEW_CURRENT](#) , [NEW_RUNTIME](#))

INTEGER [IN](#) - !(input) already opened input channel
 INTEGER [O](#) - !(input) already opened logging channel (O>0)
 REAL*8 [NEW_CURRENT](#) - !(input&output) if>0, set new current[mA]
 REAL*8 [NEW_RUNTIME](#) - !(input&output) if>0, new runtime

```

C=====
=====
C          REVISED VERSION OF HELM'S INPUT SUBROUTINE
C
C          ADDED AN ELEMENT MATRIX AND AN ELEMENT RECIRC WITH GENERAL
TRANSPORT
C          MATRIX OPERATIONS
C
C          REINJECTION POINT MARKED BY > IN MACHINE LATTICE
(CAVITY,LENS,DRIFT
C          OR MATRIX)
C
C=====
=====
* KBB: 7/23/02 - g77 disapproves of using integers for characters;
changed declarations
* KBB: 4/7/03 - enhanced CAVMAT auxillary file handling
*
* The CAVMAT file describes the transfer matrix for each cavity;
each
* entry corresponds to a CECAV card in the input file.
*
***** ****
*
```

called by: [TDBBU](#)

calls: [YESNO](#) , [RTM_PRINT](#) , [ARRAYCHECK](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#),
[TIME INFO](#), [BEAM INFO](#), [T M INFO](#), [T M](#), [DELAYS](#), [RTM INFO](#),
[BUGGY](#), [MATRIX UNITS](#), [ESTHOM](#), [DUTY](#), [HOM](#), [LASER](#),
[HOM MAP](#), [HOM INFO](#), [HOM WHEN](#), [HOM KICKS](#),
[WENT_BY_HOM](#), [PRINTING](#), [PRNT](#), [WORKING_PRINT](#),
[MATRIX FILES](#), [EXTRAPOLATE_MTX](#)

[[SRC/print_path.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_opts.par](#)] [[tdbbu_path.cmn](#)] [[tdbbu_tm.par](#)] [[tdbbu_tm.cmn](#)]

SUBROUTINE PRINT_PATH(*IO*, *TITLE*)

INTEGER *IO* - !(input) already prepared I/O channel
CHARACTER*(*) *TITLE* - !(input) title comment

```
*  
* Just print the current path to  
* prepared I/O channel #IO  
*
```

called by: [PLAN_PATH](#) , [TDBBUK](#)

commons: [PATH](#), [MISC PATH](#), [T M INFO](#), [T M](#)

[[SRC/propagate_bunch.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_bunch.cmn](#)] [[tdbbu_path.cmn](#)] [[tdbbu_1by8.par](#)] [[tdbbu_1by16.par](#)] [[tdbbu_io.par](#)]

SUBROUTINE PROPAGATE_BUNCH(*ID*, *TNOW*, *TNEXT*)

INTEGER*4 *ID* - !(input) bunch ID#
REAL*8 *TNOW* - !(output) time of bunch after this step
REAL*8 *TNEXT* - !(output) the time of bunch after next step

```
*  
* Propagate bunch#id one element - update Tnow and Tnext.  
* 6/04 KBB  
*
```

called by: [STEPK](#)

calls: [BUNCH_INFO](#) , [HOM OPERATE](#) , [TDBBU_EXT_RTM](#) ,
[RTM OPERATE](#) , [TDBBU_EXT_TM](#) , [TM OPERATE](#) ,
[ESTIMATE_T_NEXTSTEP](#) , [TDBBU_LOGBUNCH](#)

commons: [ABOUT BUNCHES](#), [ALL BUNCHES](#), [PATH](#), [MISC PATH](#),
[MATRIX UNITS](#)

[[SRC/ranf.f](#)]

REAL*8 FUNCTION RANF()

*

```

* A kludge to substitute for the system call;
* returns a random number (native size).
*
* set seed using RANSET( int8 )
*
* KBB 7/22/02
*
```

called by: [GAUSSIAN1D](#) , [GENERATE BUNCH](#) , [TDBBU EXT TM](#)

[[SRC/ranset.f](#)]

SUBROUTINE RANSET(*SEED*)

INTEGER*8 *SEED* - !input generator starting seed: should be large+& odd

```

*
* A kludge to substitute for the system call;
* sets the seed for the random number calls.
*
* KBB 7/22/02
*
```

called by: [TDBBUK](#)

[[SRC/report_recirculationtimes.f](#)]

[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu.cmn](#)] [[tdbbu_time.cmn](#)] [[tdbbu_killedstats.cmn](#)] [[tdbbu_delays.cmn](#)] [[tdbbu_io.par](#)]

SUBROUTINE REPORT_RECIRCULATIONTIMES(*PASSN* , *INFO*)

INTEGER*4 *PASSN* - !input pass#

CHARACTER*(*) *INFO* - !output description of recirculation

```

*
* Report the final info on the recirculation
* time for pass#passN
*
```

called by: [STEPK](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#),
[TIME INFO](#), [TDBBU KILLEDSTATS](#), [TDBBU LIFETIME](#),
[RECIRC STATS](#), [DELAYS](#), [MATRIX UNITS](#)

[[SRC/rms2slope.f](#)]

[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_guess.par](#)] [[tdbbu.cmn](#)] [[tdbbu_time.cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_hunt.cmn](#)] [[tdbbu_enough.cmn](#)] [[tdbbu_cuts.cmn](#)] [[tdbbu_blowup.cmn](#)] [[tdbbu_duty.cmn](#)]

SUBROUTINE RMS2SLOPE(*O* , *N* , *VALUE* , *TIMES* , *SLOPE* , *EXPONENT*)

INTEGER *O* - !(input) output channel for messages if>0

INTEGER*8 *N* - !(input) number of plotted value

REAL*8 *VALUE(*)* - !(input) plotted value

REAL*8 ***TIME_S(*)*** - !(input) times of plotted value
 REAL*8 ***SLOPE*** - !(output) slope/channel of RMS in upper half
 REAL*8 ***EXPONENT*** - !(output) exponential fit of RMS in upper half

```

*
* Given the number of and X or Y values that appear in
* the TDBBU plot file, fits a slope to the
* running RMS of the 2nd half for continuous beam, and
* for non-continuous beam, fits the extrems for each
* beam ON/OFF cycle.
*
* Both destroy the input value list. - KBB
*
*

```

called by: [TDBBU](#) , [TDBBU RESCAN PLOTS](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#),
[TIME INFO](#), [MATRIX UNITS](#), [HISTORY](#), [DEADZONE](#), [CUTS](#),
[BLOWUP](#), [DUTY](#), [HOM](#)

[[SRC/rough_estimate.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_est.cmn](#)] [[tdbbu_bug.cmn](#)]

SUBROUTINE ROUGH_ESTIMATE(***J_MA***, ***T_US***)

REAL*8 ***J_MA*** - !(output) estimate of current [mA]
 REAL*8 ***T_US*** - !(output) estimate of required runtime [uS]

```

*****
*****
*
* Given the HOM values and the energy exiting the HOMs,
* returns a rough estimate of the relevant current J and
* time required...
*
* KBB 5/29/03
*
*****
*****
*
```

called by: [TDBBU](#)

commons: [ESTHOM](#), [BUGGY](#)

[[SRC/rtm_operate.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_1by16.par](#)] [[tdbbu_delays.cmn](#)] [[tdbbu_time.cmn](#)] [[tdbbu_rtm.cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_tm.par](#)] [[tdbbu_tm.cmn](#)] [[tdbbu_opts.par](#)] [[tdbbu_bunch.par](#)]

SUBROUTINE RTM_OPERATE(***R_T_M***, ***P***, ***EID***, ***MAT***, ***BNCH***, ***RECIRC***)

REAL*8 ***R_T_M(X_X:PY_PY)*** - !input 1x16 matrix
 INTEGER*4 ***P*** - !input pass#

```

INTEGER*4 EID - !input element id#
INTEGER*4 MAT - !input matrix#
REAL*8 BNCH(BEGIN:END) - !(input&output) bunch
LOGICAL*4 RECIRC - !input whether to add recirculation delay

*
* Apply the 1x16 matrix#mat R_T_M to the bunch at pass#pass and
* element#eid; include M56 and T566 time corrections; apply
* recirculation pass time adjustment only if requested.
*
* 6/04 KBB
*

```

called by: [PROPAGATE BUNCH](#)

calls: [M4X4 PRINT](#)

commons: [DELAYS](#), [TIME INFO](#), [RTM INFO](#), [MATRIX UNITS](#),
[T M INFO](#), [T M](#)

[[SRC/rtm.print.f](#)]
[[tdbbu.defs.par](#)] [[tdbbu.generic.par](#)] [[tdbbu.rtm.cmn](#)] [[tdbbu.lby16.par](#)] [[tdbbu.io.par](#)]

SUBROUTINE RTM_PRINT(*IO*, *PASS*, *MAT*, *COMMENT*) already
opened IO channel pass# matrix# comment to identify output

```

*
* Print the RTM(pass,mat,*) array as a 4x4 array to IO channel#IO
*
```

called by: [PRECAL](#)

calls: [M4X4 PRINT](#)

commons: [RTM INFO](#), [MATRIX UNITS](#)

[[SRC/second.f](#)]

SUBROUTINE SECOND(*TO*)

REAL*8 *TO* - !input|output time in seconds

```

*
* A kludge for the system call that
* returns the number of seconds since
* midnight minus t0;
* modified so it now returns the current
* UTC time minus t0 - KBB 5/18/04
*
```

called by: [TDBBU](#)

[[SRC/skipoverlines.f](#)]
[[tdbbu.defs.par](#)] [[tdbbu.bug.cmn](#)]

SUBROUTINE **SKIPOVERLINES**(*TOCHANNEL*, *LINE*, *OK*) already opened input channel nonblank line success or failure

```
*  
* Skip over blank or comment-only input lines; return  
* the next nonblank line. Ok flags success or failure.  
*
```

called by: [TDBBU SCANMATRIXFILE](#)

commons: [BUGGY](#)

[[SRC/sortincreasing.f](#)]

SUBROUTINE **SORTINCREASING**(*N*, *ID*, *VALUE*) # of elements id#'s of elements size of each element

```
*  
* Return a increasingly sorted list of id#s and values  
*
```

called by: [TDBBU LOGBUNCH](#) , [TDBBU SCANMATRIXFILE](#)

[[SRC/stepk.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_bunch.cmn](#)] [[tdbbu.cmn](#)] [[tdbbu_time.cmn](#)] [[tdbbu_dutu.cmn](#)] [[tdbbu_path.cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_bua.cmn](#)] [[tdbbu_opts.par](#)] [[tdbbu_cuts.cmn](#)]

SUBROUTINE **STEPK**(*O*, *X_PLT*, *Y_PLT*, *NXRSLT*, *XRSLT*, *TXRSLT*,
NYRSLT, *YRSLT*, *TYRSLT*, *MAXRSLT*, *NOTE*)

INTEGER *O* - !(input) already opened logging channel (*O*>0) [native size]

INTEGER *X_PLT* - !(input) already opened plotting channels [native size]

INTEGER *Y_PLT* - !(input) already opened plotting channels [native size]

INTEGER*8 *NXRSLT* - !(output) number of X results

REAL*8 *XRSLT(*)* - !(output) X results ([cm] for XPRNT, then [MeV/c] for PXPRNT)

REAL*8 *TXRSLT(*)* - !(output) time of X results (uS)

INTEGER*8 *NYRSLT* - !(output) number of Y results

REAL*8 *YRSLT(*)* - !(output) Y results ([cm] for YPRNT, then [MeV/c] for PYPRNT)

REAL*8 *TYRSLT(*)* - !(output) time of Y results (uS)

INTEGER*8 *MAXRSLT* - !(input) maximum number of X,Y results - keep only most recent results

CHARACTER*(*) *NOTE* - !(output) brief comment

```
*  
*  
* A complete rewrite of TDBBU's step0 routine, where the whole
```

```

* philosophy has been changed.
*
* Instead of stepping the bunches as a characteristic of an
element,
* and synchronously stepping all the information from one to the
next
* element in lockstep and *assuming* 1/2 RF/cycle (clock) per
element,
* the system now has a collection of bunches that exist
independently
* of the elements.
*
* Actions still happen on clock cycles; the HOMs are updated only
* at the end of a clock cycle. Bunches cannot proceed past the
time
* of the next clock cycle, but can transverse any number of
elements
* within a clock cycle.
*
* KBB 5/28/04
*
```

called by: [TDBBU](#)

calls: [YESNO](#) , [KEEP_KILLEDSTATS](#) , [PLAN_PATH](#) ,
[GENERATE_BUNCH](#) , [ESTIMATE_T_NEXSTEP](#) , [BUNCH_INFO](#) ,
[PROPAGATE_BUNCH](#) , [TDBBU_DUMPBUNCH](#) , [HOM_OPERATE](#) ,
[HIT_WALL](#) , [STEP_RECORD](#) , [HOM_FORCE_UPDATE](#) ,
[STEP_REPORT](#) , [REPORT_RECIRCULATIONTIMES](#)

commons: [ABOUT_BUNCHES](#) , [ALL_BUNCHES](#) ,
[GENERAL_INPUT_INFO](#) , [DLOAD](#) , [R](#) , [GENERAL_INFO](#) , [TIME_INFO](#) ,
[DUTY](#) , [HOM](#) , [PATH](#) , [MISC_PATH](#) , [MATRIX_UNITS](#) , [BUGGY](#) , [CUTS](#)

```
| SRC/step\_record.f
|   [tdbbu\_defs.par] [tdbbu\_generic.par] [tdbbu.cmn] [tdbbu\_bunch.par] [tdbbu\_bunch.cmn] [tdbbu\_logged.cmn] [tdbbu\_bug.cmn] |
|   [tdbbu\_io.par] [tdbbu\_print.cmn] [tdbbu\_path.cmn]
```

SUBROUTINE [STEP_RECORD](#)([IOX](#) , [IOY](#) , [TCLOCKCOUNT](#) , [FORCEOUT](#) ,
[XRMS](#) , [YRMS](#))

INTEGER [IOX](#) - !(input) prepared I/O channel for X
INTEGER [IOY](#) - !(input) prepared I/O channel for Y
INTEGER*8 [TCLOCKCOUNT](#) - !(input) master clock [1/2 RF cycles]
LOGICAL*4 [FORCEOUT](#) - !(input) force everything to be written out
REAL*8 [XRMS](#) - !(input&output) running Xrms for 1st request, if any
REAL*8 [YRMS](#) - !(input&output) running Yrms for 1st request, if any

```

*
* Given a bunch, record any requested information. Return the
* running Xrms and Yrms for the 1st request of each, if
available;
* if not, sent them large and negative. If they enter <0, reset
* internally sums for Xrms and Yrms.
*
```

called by: [STEPK](#) , [STEP REPORT](#)

calls: [YESNO](#) , [TDBBU LOGBUNCH](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#),
[ABOUT BUNCHES](#), [ALL BUNCHES](#), [LOGGED](#), [LOGGED LOAD](#),
[BUGGY](#), [MATRIX UNITS](#), [PRINTING](#), [PRNT](#), [WORKING PRINT](#),
[PATH](#), [MISC PATH](#)

[[SRC/step_report.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_print.cmn](#)] [[tdbbu_io.par](#)]

SUBROUTINE STEP_REPORT(*X_PLT*, *Y_PLT*, *TCLK*, *NXRSLT*,
XRSLT, *TXRSLT*, *NYRSLT*, *YRSLT*, *TYRSLT*, *MAXRSLT*)

INTEGER *X_PLT* - !(input) already opened plotting channels [native size]
INTEGER *Y_PLT* - !(input) already opened plotting channels [native size]
INTEGER*8 *TCLK* - !(input) master clock [1/2 RF cycles]
INTEGER*8 *NXRSLT* - !(output) number of X results
REAL*8 *XRSLT(*)* - !(output) X results ([cm] for XPRNT, then [MeV/c] for PXPRNT)
REAL*8 *TXRSLT(*)* - !(output) time of X results (uS)
INTEGER*8 *NYRSLT* - !(output) number of Y results
REAL*8 *YRSLT(*)* - !(output) Y results ([cm] for YPRNT, then [MeV/c] for PYPRNT)
REAL*8 *TYRSLT(*)* - !(output) time of Y results (uS)
INTEGER*8 *MAXRSLT* - !(input) maximum number of X,Y results - keep only most recent results

*
* Generate final results from simulation.
*
*

called by: [STEPK](#)

calls: [STEP RECORD](#)

commons: [PRINTING](#), [PRNT](#), [WORKING PRINT](#), [MATRIX UNITS](#)

[[SRC/tdbbu_dumpbunch.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu.cmn](#)] [[tdbbu_path.cmn](#)] [[tdbbu_bunch.par](#)] [[tdbbu_bunch.cmn](#)] [[tdbbu_bunch_desc.cmn](#)] [[tdbbu_buq.cmn](#)]

SUBROUTINE TDBBU_DUMPBUNCH(*ID*) bunch ID#

*
* If the DumpSequenceID# is specified and bunch#ID# has that sequence#, dump its history to STDOUT

*

called by: [STEPK](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#), [PATH](#), [MISC PATH](#), [ABOUT BUNCHES](#), [ALL BUNCHES](#), [BUGGY](#)

[[SRC/tdbbu_ext_rtm.f](#)] [[tdbbu_1by16.par](#)] [[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu.cmn](#)] [[tdbbu_rtm.cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_matrix.cmn](#)]

SUBROUTINE TDBBU_EXT_RTM(*PASS*, *MAT*, *DPP*, *R_T_M*, *OK*)
pass# mat# fractional offset from nominal of Z momentum [-] effective
transfer matrix [cm,MeV/c] success on lookup

```
*  
* Returns a 4x4 recirculation matrix R_T_M in 1x16 form  
* for pass#p matrix#mat, extrapolating in dP/P  
*  
*      R_T_M:  
*      |X_X  X_PX  X_Y  X_PY|   | 1  2  3  4|  
*      |PX_X  PX_PX  PX_Y  PX_PY|   | 5  6  7  8|  
*      |Y_X  Y_PX  Y_Y  Y_PY|   | 9 10 11 12|  
*      |PY_X  PY_PX  PY_Y  PY_PY|   |13 14 15 16|  
*  
* 6/04 KBB
```

called by: [PROPAGATE BUNCH](#) , [TDBBU_SCANMATRIXFILE](#)

calls: [M4X4 PRINT](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#), [RTM INFO](#), [MATRIX UNITS](#), [MATRIX FILES](#), [EXTRAPOLATE MTX](#)

[[SRC/tdbbu_ext_tm.f](#)] [[tdbbu_1by8.par](#)] [[tdbbu_1by16.par](#)] [[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu.cmn](#)] [[tdbbu_time.cmn](#)] [[tdbbu_tm.par](#)] [[tdbbu_tm.cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_lasing.cmn](#)]

SUBROUTINE TDBBU_EXT_TM(*PASS*, *ID*, *DPP*, *T_M*, *OK*) current
pass# current ident# fractional momuntum offset dP/P at this point
transfer matrix for this pass# and ident# success or failure

```
*  
* Given the transfer matrixies TM (via COMMON),  
* return T_M, the 1x8 matrix for for TM(NPASS,L)  
* corrected for fractional momentum offset dPP;  
* update dPP, the momentum offset  
*  
*      T_M:  
*      |X_X  X_PX|   | 1  2|  
*      |PX_X  X_PX|   | 3  4|  
*      |Y_Y  Y_PY|   | 5  6|  
*      |PY_Y  PY_PY|   | 7  8|  
*  
* 6/04 KBB
```

called by: [PROPAGATE BUNCH](#)

calls: [RANF](#) , [GAUSSIAN1D](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#),
[TIME INFO](#), [T M INFO](#), [T M](#), [MATRIX UNITS](#), [LASER](#)

[[SRC/tdbbu.f](#)]
[[tdbbu_versioninfo.par](#)] [[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_guess.par](#)] [[tdbbu.cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_duty.cmn](#)] [
[tdbbu_1by16.par](#)] [[tdbbu_lasing.cmn](#)] [[tdbbu_delays.cmn](#)] [[tdbbu_matrix.cmn](#)]

SUBROUTINE TDBBU(*IN*, *OT*, *X_PLT*, *Y_PLT*, *NEW_J*, *NEW_RUNT*,
NXS, *XSLP*, *XEXP*, *NYS*, *YSLP*, *YEXP*, *SCAN*, *NOTE*)

INTEGER *IN* - !(input) already opened input file
INTEGER *OT* - !(input) (if>0) already opened logging IO channel
INTEGER *X_PLT* - !(input) (if>0) already opened plot IO channels
INTEGER *Y_PLT* - !(input) (if>0) already opened plot IO channels
REAL*8 *NEW_J* - !(input&output) if>0, use as new beam current; if
Scan, returns estimate
REAL*8 *NEW_RUNT* - !(input&output) if>0, use as new runtime; if
Scan, returns estimate
INTEGER*8 *NXS* - !(output) number of X data points
REAL*8 *XSLP* - !(output) slope of Xrms
REAL*8 *XEXP* - !(output) slope of ln(Xrms)
INTEGER*8 *NYS* - !(output) number of Y data points
REAL*8 *YSLP* - !(output) slope of Yrms
REAL*8 *YEXP* - !(output) slope of ln(Yrms)
LOGICAL*4 *SCAN* - !(input) whether to only scan input file for rough
estimate & return
CHARACTER*(*) *NOTE* - !(output) brief comment

*
C A VECTORIZED TWO DIMENSIONAL BEAM BREAKUP SIMULATION CODE
C=====
=====
C MULTIPASS BBU with ENERGY RECOVERY - Updated for UNICOS at NERSC
C *** Recirculation matrix now in DIMAD units
C *** THICK LENS VERSION
C *** To accomodate energy recovery scheme, specify the PASS
NUMBER at
C which ENERGY RECOVERY starts by using the variable 'IRECOV'
in aprtr.
C also, BUNCHING FREQUENCY should be changed to TWO TIMES OF
C RF FREQUENCY to enable 180 degree phase slip from
acceleration
C to deceleration in cavities.
C *** THRESHOLD CURRENT is CURR(mA) divided by IPDL(bunching
subharmonic)
C=====
=====
C *** KSTART= 1 START OUT BY FILLING MACHINE WITH ONE BEAM IN
STRUCTURE,
C THEN TWO, ETC UP TO NPASS BEAMS IN STRUCTURE KSTART= 2 START

```

OUT
C      WITH MACHINE HAVING TWO BEAMS IN STRUCTURE AND THEN BUILDING
UP
C      KSTART MUST OBVIOUSLY BE L.E. NPASS
C *** X WIGGLE =AMPX*COS(2*PI*FREQX*TIME)*RANDOM
C *** PX WIGGLE = AMPPX*COS(2*PI*FREQPX*TIME)*RANDOM
C *** IRANDX OR IRANDPX = [0],1 [NOT] RANDOM ABOVE WIGGLES ARE
ADDITIVE
C      TO HELMS INPUT X,PX
C *** RECIRCULATION MAP X=R11*X +R12*PX,PX=R21*X+R22*PX MAP OCCURS
THEN
C      WIGGLE ADDED THEN REINJECTION
C *** CEBAF CAVITY TRANSFER MATRIX CAN BE INCLUDED WITH 'CECAV'
CARD.
C      FIRST, RUN 'CEBAFCAVITY.FOR' ONLY WITH 'CECAV' CARDS TO
CREATE
C      'CAVMAT' WHICH CONTAINS TRANSFER MATRIX. THEN RUN TDDBU WITH
'CAVMAT'.
C
C      G. A. KRAFFT 17-MAY-86
C
*      July, 2004 - many modifications made; stepK algorithm
replaced
*                  step0 algorithm. K.B.Beard
*
*
```

called by: [TDBBUK](#)

calls: [YESNO](#) , [SECOND](#) , [WIPE ALL](#) , [CLOCKK](#) , [PRECAL](#) ,
[PLAN PATH](#) , [TDDBU SCANMATRIXFILE](#) , [ROUGH ESTIMATE](#) ,
[VERIFY SPACE](#) , [STEPK](#) , [HOM SUMMARY](#) , [RMS2SLOPE](#)

commons: [VERSIONINFO](#) , [GENERAL INPUT INFO](#) , [DLOAD](#) , [R](#) ,
[GENERAL INFO](#) , [MATRIX UNITS](#) , [DUTY](#) , [HOM](#) , [LASER](#) , [DELAYS](#) ,
[MATRIX FILES](#) , [EXTRAPOLATE MTX](#)

```
[SRC/tdbbuk.f]
[tdbbu_versioninfo.par] [tdbbu_defs.par] [tdbbu_generic.par] [tdbbu_guess.par] [tdbbu_cmn] [tdbbu_io.par] [tdbbu_enough.cmn] [
tdbbu_cuts.cmn] [tdbbu_bug.cmn] [tdbbu_axes.cmn] [tdbbu_est.cmn] [tdbbu_duty.cmn] [tdbbu_blowup.cmn] [tdbbu_1by16.par] [
tdbbu_lasing.cmn] [tdbbu_1by8.par] [tdbbu_rtm.cmn] [tdbbu_delays.cmn] [tdbbu_time.cmn] [tdbbu_matrix.cmn] [tdbbu_opts.cmn]
```

INTEGER FUNCTION TDBBUK()

```

*
* A simple interface to the TDDBU program;
* uses the KWWRAP interface. KBB 25jul02
*
```

called by: [main](#)

calls: [YESNO](#) , [RANSET](#) , [CLEANLIST](#) , [TDDBU RESCAN PLOTS](#) , [TDDBU](#) ,
[PRINT PATH](#) , [VERIFY SPACE](#) , [GUESSNEXT](#) , [GUESSSHOW](#)

commons: [VERSIONINFO](#) , [GENERAL INPUT INFO](#) , [DLOAD](#) , [R](#) ,
[GENERAL INFO](#) , [MATRIX UNITS](#) , [DEADZONE](#) , [CUTS](#) , [BUGGY](#) ,
[ONLYCONSIDER](#) , [ESTHOM](#) , [DUTY](#) , [HOM](#) , [BLOWUP](#) , [LASER](#) , [RTM INFO](#),

DELAYS, TIME INFO, MATRIX FILES, EXTRAPOLATE mtx, IFHITWALL

[[SRC/tdbbu_logbunch.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_bunch_desc.cmn](#)] [[tdbbu_logged.cmn](#)] [[tdbbu_io.par](#)]

SUBROUTINE TDBBU_LOGBUNCH(*PASSN*, *EID*, *BNCH*, *TIMESORT*)

INTEGER*4 *PASSN* - !(input) pass#
INTEGER*4 *EID* - !(input) element ID#
REAL*8 *BNCH(BEGIN:END)* - !(input) bunch description
LOGICAL*4 *TIMESORT* - !(input) sort all entries by total time instead

*
* Log the bunch *bnch* at *pass#passN*, element #*eID*,
* unless requested to sort all entries by time instead.
* In that case, only sort for *pass,element* if both within
* range; otherwise, sort all.
*

called by: [PROPAGATE BUNCH](#) , [STEP RECORD](#)

calls: [SORTINCREASING](#)

commons: [LOGGED](#), [LOGGED LOAD](#), [MATRIX UNITS](#)

[[SRC/tdbbu_recall_pz.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_cmn](#)] [[tdbbu_path.cmn](#)]

SUBROUTINE TDBBU_RECALL_PZ(*PASSID*, *MTXID*, *RTMID*, *P_Z*,
OK) pass# nonrecirc. matrix ID# (0=unused) recirc. matrix ID#
(0=unused) momentum success or failure

*
* Given a pass#, and either a (nonrecirculation) matrix id#MTXid
* or a recirculation matrix id#RTMid, return the nominal momentum,
* if possible, and whether successfull or not.
*
*

called by: [TDBBU_SCANMATRIXFILE](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#), [PATH](#),
[MISC PATH](#)

[[SRC/tdbbu_rescan_plots.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_io.par](#)] [[tdbbu_bug.cmn](#)]

SUBROUTINE TDBBU_RESCAN_PLOTS(*O*, *FILE*, *J*, *PTS*,
SLP_RMS, *SLP_LN*, *OK*) opened channel for messages name of plot file
to scan stated current (if found, 0 otherwise) # of points found slope of
the running RMS values slope of the log of the running RMS values
success or failure

```
*  
* Rescan the last lines of the plot file, find the slope of the  
* RMS and the slope of the log of the RMS.  
*
```

called by: [TDBBUK](#)

calls: [RMS2SLOPE](#)

commons: [MATRIX UNITS](#), [BUGGY](#)

```
[SRC/tdbbu_scanmatrixfile.f]  
[tdbbu_defs.par] [tdbbu_generic.par] [tdbbu_1by16.par] [tdbbu_6x6.par] [tdbbu_bug.cmn] [tdbbu_io.par] [tdbbu_rtm.cmn] [  
tdbbu_matrix.cmn] [tdbbu_delays.cmn]
```

SUBROUTINE TDBBU_SCANMATRIXFILE(*O*, *OK*) already prepared
log channel report success or failure

```
*  
* Open and scan file for a list of recirculation  
* matricies for various momentum offsets for  
* pass# and matrix#. This file comes from DIMAD  
* and is in DIMAD units!  
*  
* Afterward, updates the default RTM recirculation  
* matrix to the extrapolated dP/P=0 case.  
* 6/9/04 KBB  
*
```

called by: [TDBBU](#)

calls: [YESNO](#) , [SKIPOVERLINES](#) , [SORTINCREASING](#) ,
[TDBBU RECALL PZ](#) , [M4X4 PRINT](#) , [TDBBU EXT RTM](#)

commons: [BUGGY](#), [MATRIX UNITS](#), [RTM INFO](#), [MATRIX FILES](#),
[EXTRAPOLATE MTX](#), [DELAYS](#)

```
[SRC/timeorder.f]  
[tdbbu_generic.par] [tdbbu_defs.par] [tdbbu_io.par] [tdbbu_opts.par]
```

SUBROUTINE TIMEORDER(*NIDS*, *IDS*, *TIMES*, *WARNING*)

```
INTEGER*4 NIDS - !(input&output) number of elements  
INTEGER*4 IDS(*) - !(input&output) sequential#s  
REAL*8 TIMES(*) - !(input&output) values  
LOGICAL*4 WARNING - !(output) warning that duplicate IDs found
```

```
*  
* Given lists of ID#s and times, reshuffled the ids  
* for increasing times and remove any duplicate ids,  
* keeping only the earliest or latest, as specified.  
*
```

called by: [HOM OPERATE](#)

calls: [YESNO](#)

commons: [MATRIX UNITS](#)

[[SRC/tm_operate.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_1by8.par](#)] [[tdbbu_tm.par](#)] [[tdbbu_tm.cmn](#)] [[tdbbu_rtm.cmn](#)] [[tdbbu_time.cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_bunch.par](#)] [[tdbbu_opts.par](#)]

SUBROUTINE TM_OPERATE(*T_M*, *P*, *E*, *BNCH*)

REAL*8 *T_M(X_X;PY_PY)* - !(input) transfer matrix

INTEGER*4 *P* - !(input) current pass#

INTEGER*4 *E* - !(input) element id#

REAL*8 *BNCH(BEGIN:END)* - !(input&output) bunch

*
* Operates the 1x8 TM matrix on the bunch at pass#p,
* element#e; updates time, and time & momentum offset
* vectors.
*
* 6/04 KBB
*

called by: [ESTIMATE T NEXTSTEP](#) , [PROPAGATE BUNCH](#)

calls: [YESNO](#)

commons: [T M INFO](#), [T M](#), [RTM INFO](#), [TIME INFO](#), [MATRIX UNITS](#)

[[SRC/verify_space.f](#)]
[[tdbbu_defs.par](#)] [[tdbbu_generic.par](#)] [[tdbbu_guess.par](#)] [[tdbbu.cmn](#)] [[tdbbu_time.cmn](#)] [[tdbbu_io.par](#)] [[tdbbu_duty.cmn](#)] [[tdbbu_print.cmn](#)]

SUBROUTINE VERIFY_SPACE(*RUNTIME_US*, *ENOUGHSPACE*, *ERR*)

REAL*8 *RUNTIME_US* - !(input) runtime in uS

LOGICAL*4 *ENOUGHSPACE* - !(output) whether there is enough
memory

CHARACTER*(*) *ERR* - !(output) return a comment, if required

*
* Verifies that there is sufficient space allocated
* to run the simulation as specified
*
*

called by: [TDBBU](#) , [TDBBUK](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#),
[TIME INFO](#), [MATRIX UNITS](#), [DUTY](#), [HOM](#), [PRINTING](#), [PRNT](#),
[WORKING PRINT](#)

[[SRC/whereat.f](#)]

SUBROUTINE WHEREAT(*DECLARED_AS*, *WHERE*, *STRING*, *LSTRING*)

CHARACTER* (*) ***DECLARED_AS*** - !(input) variable declaration
 INTEGER*4 ***WHERE*** - !(input) 1-dimensional location
 CHARACTER* (*) ***STRING*** - !(output) conventional string NAME(i,...,k)
 INTEGER*4 ***LSTRING*** - !(output) length of string

```

*
*      Given a FORTRAN variable declaration string and
*      its 1D index, returns a string in the conventional
*      form.
*
*      For example declared_as="V(10,3,-1:1)" and
*      where=5 --> string="(5,1,-1)"
*
*
*
```

called by: [ARRAYCHECK](#)

calls: [WHEREINDEXMAP](#)

[[SRC/whereindexmap.f](#)]
 SUBROUTINE WHEREINDEXMAP(***NDIM***, ***DIMLO***, ***DIMHI***, ***WHERE***,
OK, ***STR***)

INTEGER*4 ***NDIM*** - !(input) number of dimensions
 INTEGER*4 ***DIMLO(*)*** - !(input) low end for each dimension
 INTEGER*4 ***DIMHI(*)*** - !(input) high end for each dimension
 INTEGER*4 ***WHERE*** - !(input) 1-dimensional location
 LOGICAL*4 ***OK*** - !(output) within boundries
 CHARACTER* (*) ***STR*** - !(output) conventional string(i,...,k)

```

*
*      Given a variable's FORTRAN declared dimensionality
*      and its 1D index, returns a string str in the
*      conventional form - KBB
*
*      For example, if V(10,3,-1:1) -> Ndim=3
*                  dimLO(1)=1,dimLO(2)=1,dimLO(3)=-1
*                  dimHI(1)=10,dimHI(2)=3,dimHI(3)=1
*      and where= 5 --> str="(5,1,-1)"
*
*
```

called by: [WHEREAT](#)

[[SRC/wipe_all.f](#)]
[tdbbu_defs.par](#) | [tdbbu_generic.par](#) | [tdbbu_cmn](#) | [tdbbu_tm.par](#) | [tdbbu_tm.cmn](#) | [tdbbu_rtm.cmn](#) | [tdbbu_duty.cmn](#) | [tdbbu_delays.cmn](#) |
[tdbbu_io.par](#) | [tdbbu_print.cmn](#)

SUBROUTINE WIPE_ALL()

```

*
*      Zero out the DLOAD, R, PRNT, T_M, and
*      general_input_info COMMONS.
```

*

called by: [TDBBU](#)

calls: [HOM RESET ALL](#) , [ERASE ALL BUNCHES](#) , [ERASE LOGGED](#) ,
[KEEP KILLEDSTATS](#)

commons: [GENERAL INPUT INFO](#), [DLOAD](#), [R](#), [GENERAL INFO](#),
[T M INFO](#), [T M](#), [RTM INFO](#), [DUTY](#), [HOM](#), [DELAYS](#), [MATRIX UNITS](#),
[PRINTING](#), [PRNT](#), [WORKING PRINT](#)

[[SRC/yesno.f](#)]

CHARACTER*3 FUNCTION YESNO(*MAYBE*)

LOGICAL*4 *MAYBE* - !input whether True or False

*
* Returns either "YES" or "NO " based on Maybe
*

called by: [HOM OPERATE](#) , [PLAN PATH](#) , [PRECAL](#) , [STEPK](#) ,
[STEP RECORD](#) , [TDBBU](#) , [TDBBUK](#) , [TDBBU SCANMATRIXFILE](#) ,
[TIMEORDER](#) , [TM OPERATE](#)

[[tdbbuk_main.c](#)]

int [**main**](#)(int *argc* , char **argv*[])

int *argc* -
char **argv*[] -

Kevin's WRAPper
http://wims3.larc.nasa.gov/~beard/KWRAP/
defined during compilation

calls: [TDBBUK](#)

COMMON blocks

COMMON block name	group#1 routines
--------------------------	-----------------------------------

ABOUT_BUNCH ES

BUNCH INFO
ERASE ALL BUNCHES HIT WALL
TDBBU CHECKLOSSSES
HOM OPERATE
KEEP KILLEDSTATS
PROPAGATE BUNCH STEPK
STEP RECORD
TDBBU DUMPBUNCH

INTEGER*8 SEQUENTIALBUNCHCOUNT
INTEGER*8 BUNCHESINTODUMP
INTEGER*8 BUNCHESLOST
INTEGER*4 NBUNCHES
INTEGER*4 OLDESTBUNCH

ALL_BUNCHES

BUNCH INFO
ERASE ALL BUNCHES HIT WALL
TDBBU CHECKLOSSSES
HOM OPERATE
KEEP KILLEDSTATS
PROPAGATE BUNCH STEPK
STEP RECORD
TDBBU DUMPBUNCH

REAL*8 BUNCH(BEGIN:END,LOWEST_BUNCHID:HIGHEST_BUNCHID)
INTEGER*8 BUNCH_SEQUENCE(LOWEST_BUNCHID:HIGHEST_BUNCHID)
INTEGER*4 BUNCH_AT_STEP(LOWEST_BUNCHID:HIGHEST_BUNCHID)
REAL*8 BUNCH_TRECIRC(0:MPAS,LOWEST_BUNCHID:HIGHEST_BUNCHID)
INTEGER*4 BUNCH_DEATHSTEP(LOWEST_BUNCHID:HIGHEST_BUNCHID)

BEAM_INFO

PLAN PATH PRECAL

REAL*8 BEAM_UNAVERAGED_MA
INTEGER*8 BEAM_SUBHARMONIC_NUMBER
INTEGER*4 DUMPATENDOFELEMENT

BLOWUP

HIT WALL RMS2SLOPE TDBBUK

REAL*8 APERATUREFORLOSS

BUGGY

HOM OPERATE PLAN PATH PRECAL
ROUGH ESTIMATE SKIPOVERLINES
STEPK STEP RECORD
TDBBU DUMPBUNCH TDBBUK
TDBBU RESCAN PLOTS
TDBBU SCANMATRIXFILE

LOGICAL*8 DEBUG_GLOBAL
INTEGER*8 BUGGINESS

	INTEGER*8 INPROGRESSINTERVAL
	INTEGER*8 VERBOSITY
	LOGICAL*8 VERBOSE
	LOGICAL*8 IGNOREERRORS
	INTEGER*8 DUMPSEQUENCEID
CUTS	<u>GUESSNEXT RMS2SLOPE STEPK</u> <u>TDBBUK</u>
	REAL*8 DISCARDINITIALFRACTION
	REAL*8 CLOSEENOUGHFRACTION
	REAL*8 FLUCTUATIONBENEATHNOTICE
	LOGICAL*4 ALLOWQUITEARLY
DEADZONE	<u>GUESSNEXT RMS2SLOPE TDBBUK</u>
	REAL*8 DEADBAND(LO:HII)
	REAL*8 DUTYLOOKSSTABLE
DELAYS	<u>ESTIMATE T NEXTSTEP PRECAL</u> <u>REPORT RECIRCULATIONTIMES</u> <u>RTM OPERATE TDBBU TDBBUK</u> <u>TDBBU SCANMATRIXFILE</u> <u>WIPE ALL</u>
	INTEGER*8 ITD(MMAT,MPAS)
	INTEGER*8 ITREC(MPAS)
	INTEGER*8 IBUFL(0:MMAT,MPAS)
	INTEGER*8 IBUFU(0:MMAT,MPAS)
	REAL*8 ADJUST_DELAY(MPAS)
	REAL*8 RTM_M56(0:MAX_ORDER_M56,MMAT,MPAS)
	INTEGER*4 RTMO_M56(MMAT,MPAS)
	LOGICAL*4 RTM_M56_COMMANDED(MMAT,MPAS)
DLOAD	<u>GENERATE BUNCH GUESSNEXT</u> <u>GUESSSHOW KEEP KILLEDSTATS</u> <u>PLAN PATH PRECAL</u> <u>REPORT RECIRCULATIONTIMES</u> <u>RMS2SLOPE STEPK STEP RECORD</u> <u>TDBBU DUMPBUNCH</u> <u>TDBBU EXT RTM TDBBU EXT TM</u> <u>TDBBU TDBBUK</u> <u>TDBBU RECALL PZ VERIFY SPACE</u> <u>WIPE ALL</u>
	INTEGER*8 IPDL
	REAL*8 FCUR(MMOD)

DUTY	<u>GUESSNEXT</u> <u>GUESSSHOW</u> <u>PRECAL</u> <u>RMS2SLOPE</u> <u>STEPK</u> <u>TDBBU</u> <u>TDBBUK</u> <u>VERIFY</u> <u>SPACE</u> <u>WIPE</u> <u>ALL</u>
	REAL*8 TOTAL_RUNTIME_US
	REAL*8 REQ_DUTYONUS
	REAL*8 REQ_DUTYOFFUS
	REAL*8 DUTYONUS
	REAL*8 DUTYOFFUS
	LOGICAL*4 DUTYCYCLEREQUEST
	LOGICAL*4 DUTYCYCLEUNREQUEST
	LOGICAL*4 DUTYCYCLEENABLED
	LOGICAL*4 ADJUSTINTERVALIFNEEDED
ESTHOM	<u>PRECAL</u> <u>ROUGH</u> <u>ESTIMATE</u> <u>TDBBUK</u>
	INTEGER*8 N_ESTHOM
	REAL*8 TRECYLE_US
	REAL*8 M12_RECYLE
	REAL*8 EXITENERGY_ESTHOM(MAX_ESTHOMS)
	REAL*8 ROVERQ_ESTHOM(MAX_ESTHOMS)
	REAL*8 FREQ_ESTHOM(MAX_ESTHOMS)
	REAL*8 Q_ESTHOM(MAX_ESTHOMS)
	LOGICAL*4 UPDATE_ESTHOM(MAX_ESTHOMS)
	INTEGER*8 BEAM_HARMONIC
EXTRAPOLATE_MTX	<u>PRECAL</u> <u>TDBBU</u> <u>EXT</u> <u>RTM</u> <u>TDBBU</u> <u>TDBBUK</u> <u>TDBBU</u> <u>SCANMATRIXFILE</u>
	REAL*8 EXTRAPMTX(X_PY_PY,MAXEXTRAPMTX,MMAT,MPAS)
	REAL*8 EXTRAPMTX_DPP(MAXEXTRAPMTX,MMAT,MPAS)
	REAL*8 EXTRAPPZ(MMAT,MPAS)
	INTEGER*4 NEXTRAPMTX(MMAT,MPAS)
	LOGICAL*4 EXTRAPMTXRECIRCULATION(MMAT,MPAS)
	<u>GENERATE</u> <u>BUNCH</u> <u>GUESSNEXT</u> <u>GUESSSHOW</u> <u>KEEP</u> <u>KILLEDSTATS</u> <u>PLAN</u> <u>PATH</u> <u>PRECAL</u> <u>REPORT</u> <u>RECIRCULATIONTIMES</u> <u>RMS2SLOPE</u> <u>STEPK</u> <u>STEP</u> <u>RECORD</u> <u>TDBBU</u> <u>DUMPBUNCH</u> <u>TDBBU</u> <u>EXT</u> <u>RTM</u> <u>TDBBU</u> <u>EXT</u> <u>TM</u> <u>TDBBU</u> <u>TDBBUK</u> <u>TDBBU</u> <u>RECALL</u> <u>PZ</u> <u>VERIFY</u> <u>SPACE</u> <u>WIPE</u> <u>ALL</u>
GENERAL_INFO	INTEGER*4 HIGHEST_ELEMENT_FOUND
	INTEGER*4 HIGHEST_PASS_FOUND

GENERAL_INPUT_INFO

GENERATE BUNCH GUESSNEXT
GUESSSHOW KEEP KILLEDSTATS
PLAN PATH PRECAL
REPORT RECIRCULATIONTIMES
RMS2SLOPE STEPK STEP RECORD
TDBBU DUMPBUNCH
TDBBU EXT RTM TDBBU EXT TM
TDBBU TDBBUK
TDBBU RECALL PZ VERIFY SPACE
WIPE ALL

INTEGER*8 NTHLINE

CHARACTER*80 WHATSHAPPENING

HISTORY

GUESSNEXT GUESSREPORT
GUESSSHOW RMS2SLOPE

INTEGER*8 N_HISTORY

REAL*4 PTS_HISTORY(NONE:MAX_HISTORY,X:Y)

REAL*4 JHISTORY(NONE:MAX_HISTORY)

REAL*4 SLPHISTORY(NONE:MAX_HISTORY,X:Y)

REAL*4 EXPHistory(NONE:MAX_HISTORY,X:Y)

CHARACTER*1 HISTORY(NONE:MAX_HISTORY,X:Y)

CHARACTER*80 CURRENT_STATUS(X:Y)

HOM

GUESSNEXT GUESSSHOW PRECAL
RMS2SLOPE STEPK TDBBU
TDBBUK VERIFY SPACE WIPE ALL

REAL*8 HOM_W(MELM)

REAL*8 HOM_Q(MELM)

HOM_INFO

HOM FORCE UPDATE

HOM OPERATE HOM RESET ALL

HOM SUMMARY PRECAL

REAL*8 CF(MELM,CF_FROM:CF_TO)

REAL*8 CF_INFO(MELM,CFINFO_ROVERQ:CFINFO_FREQ)

REAL*8 F(MELM)

REAL*8 G(MELM)

HOM_KICKS

HOM FORCE UPDATE

HOM OPERATE HOM RESET ALL

HOM SUMMARY PRECAL

REAL*8 SUMXKICK2(MELM)

REAL*8 SUMYKICK2(MELM)

REAL*8 KICKS(MELM)

HOM_MAP	<u>HOM FORCE UPDATE</u> <u>HOM OPERATE HOM RESET ALL</u> <u>HOM SUMMARY PRECAL</u> INTEGER*8 NCAV INTEGER*8 ICAPV(MELM) INTEGER*4 ELEMENTTOHOM(MELM)
HOM_WHEN	<u>HOM FORCE UPDATE</u> <u>HOM OPERATE HOM RESET ALL</u> <u>HOM SUMMARY PRECAL</u> REAL*8 LASTT(MELM) INTEGER*8 LASTTSEQ(MELM) INTEGER*8 LASTTCLK(MELM) INTEGER*4 LASTTSTEP(MELM) INTEGER*8 LASTTACT(MELM)
IFHITWALL	<u>GUESSNEXT HIT WALL</u> <u>TDBBU CHECKLOSSSES TDBBUK</u> REAL*8 WAYOUTSIDEAPERATURE REAL*8 ABORTONFRACTIONALLOSS LOGICAL*4 ABORTRUNIFANYLOST LOGICAL*4 HUNTFORBLOWUP
LASER	<u>PRECAL TDBBU EXT TM TDBBU</u> <u>TDBBUK</u> REAL*8 LASER_DPP LOGICAL*4 LASERREQUESTED LOGICAL*4 LASERPRESENT LOGICAL*4 LASERON LOGICAL*4 LASERINITIALIZED INTEGER*4 LASERPASN INTEGER*4 LASERELEMENTN INTEGER*4 LASERLINEN
LOGGED	<u>ERASE LOGGED STEP RECORD</u> <u>TDBBU LOGBUNCH</u> REAL*8 MOSTRECENT(LOGFIRST:LOGLAST,0:MELM,0:MPAS,0:LOGSPACE-1) REAL*8 TMOSTRECENT(LOGFIRST:LOGLAST,0:MELM,0:MPAS,0:LOGSPACE-1) INTEGER*8 NUNLOGGEDYET(LOGFIRST:LOGLAST,0:MELM,0:MPAS) INTEGER*4 UNLOGGEDIDX(LOGFIRST:LOGLAST,0:MELM,0:MPAS)
LOGGED_LOAD	<u>ERASE LOGGED STEP RECORD</u> <u>TDBBU LOGBUNCH</u> REAL*8 CNT_UNLOGGED(LOGFIRST:LOGLAST,0:MELM,0:MPAS) REAL*8 AVG_UNLOGGED(LOGFIRST:LOGLAST,0:MELM,0:MPAS) REAL*8 PEAK_UNLOGGED(LOGFIRST:LOGLAST,0:MELM,0:MPAS)
MATRIX_FILES	<u>PRECAL TDBBU EXT RTM TDBBU</u> <u>TDBBUK TDBBU SCANMATRIXFILE</u>

	CHARACTER*256 MATRIXFILE(MMAT)
	INTEGER*4 MATRIXFILEMAT(MMAT)
	INTEGER*4 MATRIXFILEPASS(MMAT)
	INTEGER*4 NMATRIXFILE
	LOGICAL*4 MATRIXFILERECIRCULATION(MMAT)
MATRIX_UNITS	<u>ARRAYCHECK</u>
	<u>ERASE_ALL_BUNCHES</u>
	<u>ERASE_LOGGED</u>
	<u>ESTIMATE_T_NEXTSTEP</u>
	<u>GENERATE_BUNCH_GUESSNEXT</u>
	<u>GUESSREPORT_GUESSSHOW</u>
	<u>HIT_WALL_HOM_FORCE_UPDATE</u>
	<u>HOM_OPERATE_HOM_RESET_ALL</u>
	<u>HOM_SUMMARY</u>
	<u>KEEP_KILLEDSTATS_PLAN_PATH</u>
	<u>PRECAL_PROPAGATE_BUNCH</u>
	<u>REPORT_RECIRCULATIONTIMES</u>
	<u>RMS2SLOPE_RTM_OPERATE</u>
	<u>RTM_PRINT_STEPK_STEP_RECORD</u>
	<u>STEP_REPORT_TDDBU_EXT_RTM</u>
	<u>TDDBU_EXT_TM_TDDBU_TDDBUK</u>
	<u>TDDBU_LOGBUNCH</u>
	<u>TDDBU_RESCAN_PLOTS</u>
	<u>TDDBU_SCANMATRIXFILE</u>
	<u>TIMEORDER_TM_OPERATE</u>
	<u>VERIFY_SPACE_WIPE_ALL</u>
MISC_PATH	LOGICAL*8 MATRIX_SYSTEM(UNKNOWN:DIMAD)
	LOGICAL*8 CAVITY_SYSTEM(UNKNOWN:DIMAD)
	CHARACTER*8 NAME_SYSTEM(UNKNOWN:DIMAD)
	LOGICAL*8 USINGCAVMAT
	<u>BUNCH_INFO</u>
	<u>ESTIMATE_T_NEXTSTEP_HIT_WALL</u>
	<u>TDDBU_SORTDEATHSTATS</u>
	<u>HOM_OPERATE_PLAN_PATH</u>
	<u>PRINT_PATH_PROPAGATE_BUNCH</u>
	<u>STEPK_STEP_RECORD</u>
	<u>TDDBU_DUMPBUNCH</u>
	<u>TDDBU_RECALL_PZ</u>
	INTEGER*4 LASTELEMENTSPECIFIED

ONLYCONSIDER	GUESSNEXT TDDBUK
	LOGICAL*4 ONLYXCONSIDER
	LOGICAL*4 ONLYYCONSIDER
PATH	<u>BUNCH INFO</u> <u>ESTIMATE T NEXTSTEP HIT WALL</u> <u>TDBBU SORTDEATHSTATS</u> <u>HOM OPERATE PLAN PATH</u> <u>PRINT PATH PROPAGATE BUNCH</u> <u>STEPK STEP RECORD</u> <u>TDBBU DUMPBUNCH</u> <u>TDBBU RECALL PZ</u>
	INTEGER*4 NOPERATIONS
	INTEGER*4 OPERATIONTODO(PASS:HOM_ACTION,0:MPAS*MELM)
	INTEGER*8 OPERATIONALDEATHS(0:MPAS*MELM)
PRINTING	<u>PRECAL STEP RECORD</u> <u>STEP REPORT VERIFY SPACE</u> <u>WIPE ALL</u>
	INTEGER*8 IFP
	INTEGER*8 NPRT(MELM)
	INTEGER*8 NGPRNT
	INTEGER*8 ITPRNT
PRNT	<u>PRECAL STEP RECORD</u> <u>STEP REPORT VERIFY SPACE</u> <u>WIPE ALL</u>
	INTEGER*8 NPX
	INTEGER*8 MXPRNT(MAX_PRNT_SIZE)
	INTEGER*8 LXPRNT(MAX_PRNT_SIZE)
	INTEGER*8 IXPRNT(MAX_PRNT_SIZE)
	INTEGER*8 NPPX
	INTEGER*8 MPXPRNT(MAX_PRNT_SIZE)
	INTEGER*8 LPXPRNT(MAX_PRNT_SIZE)
	INTEGER*8 IPXPRNT(MAX_PRNT_SIZE)
	INTEGER*8 NPY
	INTEGER*8 MYPRNT(MAX_PRNT_SIZE)
	INTEGER*8 LYPRNT(MAX_PRNT_SIZE)
	INTEGER*8 IYPRNT(MAX_PRNT_SIZE)
	INTEGER*8 NPPY
	INTEGER*8 MPYPRNT(MAX_PRNT_SIZE)
	INTEGER*8 LPYPRNT(MAX_PRNT_SIZE)
	INTEGER*8 IPYPRNT(MAX_PRNT_SIZE)

R

GENERATE BUNCH GUESSNEXT
GUESSSHOW KEEP KILLEDSTATS
PLAN PATH PRECAL
REPORT RECIRCULATIONTIMES
RMS2SLOPE STEPK STEP RECORD
TDBBU DUMPBUNCH
TDBBU EXT RTM TDBBU EXT TM
TDBBU TDBBUK
TDBBU RECALL PZ VERIFY SPACE
WIPE ALL

REAL*8 PZPASS(MPAS,MELM)

REAL*8 AMPX(MPAS)

REAL*8 FREQX(MPAS)

INTEGER*8 IRANDX(MPAS)

REAL*8 AMPPX(MPAS)

REAL*8 FREQPX(MPAS)

INTEGER*8 IRANDPX(MPAS)

REAL*8 AMPY(MPAS)

REAL*8 FREQY(MPAS)

INTEGER*8 IRANDY(MPAS)

REAL*8 AMPY(MPAS)

REAL*8 FREQY(MPAS)

INTEGER*8 IRANDPY(MPAS)

INTEGER*8 ISTART

INTEGER*8 NPASS

INTEGER*8 NMB

INTEGER*8 IRECIRC

INTEGER*8 IRECFLG

INTEGER*8 IRECPRT

INTEGER*8 ITIMES

INTEGER*8 KSTART

INTEGER*8 ITCAV

INTEGER*8 ITON

INTEGER*8 ITOFF

INTEGER*8 ITFL

REAL*8 CURR

REAL*8 XOFF

REAL*8 PXOFF

REAL*8 YOFF

REAL*8 PYOFF

INTEGER*8 NMAT

RECIRC_STATS

KEEP_KILLEDSTATS
REPORT RECIRCULATIONTIMES

REAL*8 RECIRC_SUMT(MPAS)

REAL*8 RECIRC_SUMT2(MPAS)

INTEGER*8 RECIRC_NSUMS(MPAS)

	REAL*8 RECIRCTAVG(MPAS)
	REAL*8 RECIRCTRMS(MPAS)
RTM_INFO	<u>PRECAL RTM OPERATE RTM PRINT</u> <u>TDBBU EXT RTM TDBBUK</u> <u>TDBBU SCANMATRIXFILE</u> <u>TM OPERATE WIPE ALL</u>
	REAL*8 RTM(MPAS,MMAT,16)
TDBBU_KILLED STATS	<u>KEEP KILLEDSTATS</u> <u>REPORT RECIRCULATIONTIMES</u>
	INTEGER*8 NKILLED
	REAL*8 KILLED_SUM(BEGIN:END)
	REAL*8 KILLED_SUM2(BEGIN:END)
	REAL*8 KILLED_AVG(BEGIN:END)
	REAL*8 KILLED_RMS(BEGIN:END)
TDBBU_LIFETIME	<u>KEEP KILLEDSTATS</u> <u>REPORT RECIRCULATIONTIMES</u>
	REAL*8 LIFETIME_SUM
	REAL*8 LIFETIME_SUM2
	REAL*8 LIFETIME_AVG
	REAL*8 LIFETIME_RMS
TIME_INFO	<u>BUNCH INFO</u> <u>ESTIMATE T NEXTSTEP</u> <u>HOM OPERATE PLAN PATH PRECAL</u> <u>REPORT RECIRCULATIONTIMES</u> <u>RMS2SLOPE RTM OPERATE STEPK</u> <u>TDBBU EXT TM TDBBUK</u> <u>TM OPERATE VERIFY SPACE</u>
	REAL*8 RF_FREQUENCY_MHZ
	REAL*8 BUNCHING_SUBHARMONIC
	REAL*8 CLOCK_US
	INTEGER*8 CLOCKNOW
	REAL*8 CURRENTTIME_US
T_M	<u>ESTIMATE T NEXTSTEP</u> <u>HOM OPERATE PLAN PATH PRECAL</u> <u>PRINT PATH RTM OPERATE</u> <u>TDBBU EXT TM TM OPERATE</u> <u>WIPE ALL</u>
	REAL*8 TM(MPAS,MELM,8)
	REAL*8 Z(MELM)
	REAL*8 PZ(MELM+1)

	INTEGER*8 MATP(MMAT)
	REAL*8 TB
	REAL*8 APRTR
	INTEGER*8 NLMNTS
	INTEGER*8 NMA
T_M_INFO	<u>ESTIMATE T NEXTSTEP</u> <u>HOM OPERATE PLAN PATH PRECAL</u> <u>PRINT PATH RTM OPERATE</u> <u>TDBBU EXT TM TM OPERATE</u> <u>WIPE ALL</u>
	INTEGER*4 TMTYPE(MPAS,MELM)
	REAL*8 TMDEGOFFCREST(MPAS,MELM)
	REAL*8 TMLLENGTH(MELM)
	REAL*8 TMDKE(MPAS,MELM)
	REAL*8 TMENTERP(MPAS,MELM)
	REAL*8 TMEXITP(MPAS,MELM)
	CHARACTER*12 TMTYPE_DESC(TMTYPE_UNSPECIFIED:TMTYPE_HOM)
	CHARACTER*80 TMLINE(MELM)
	INTEGER*4 TMLINENTH(MELM)
VERSIONINFO	<u>TDBBU TDBBUK</u>
	CHARACTER*512 FULLVERSIONINFO
WENT_BY_HOM	<u>HOM FORCE UPDATE</u> <u>HOM OPERATE HOM RESET ALL</u> <u>HOM SUMMARY PRECAL</u>
	REAL*8 WENTBY_TIME(BUNCHSPACE,MELM)
	INTEGER*4 WENTBY_ID(BUNCHSPACE,MELM)
	INTEGER*4 NWENTBY(MELM)
WORKING_PRIN T	<u>PRECAL STEP RECORD</u> <u>STEP REPORT VERIFY SPACE</u> <u>WIPE ALL</u>
	REAL*8 XPRNT(MAX_PRNT_SIZE,NPLOT)
	REAL*8 PXPRNT(MAX_PRNT_SIZE,NPLOT)
	REAL*8 YPRNT(MAX_PRNT_SIZE,NPLOT)
	REAL*8 PYPRNT(MAX_PRNT_SIZE,NPLOT)
	REAL*8 TXPRNT(MAX_PRNT_SIZE,NPLOT)
	REAL*8 TPXPRNT(MAX_PRNT_SIZE,NPLOT)
	REAL*8 TYPRNT(MAX_PRNT_SIZE,NPLOT)
	REAL*8 TPYPRNT(MAX_PRNT_SIZE,NPLOT)
	INTEGER*4 NUMPX(MAX_PRNT_SIZE)
	INTEGER*4 NUMPPX(MAX_PRNT_SIZE)
	INTEGER*4 NUMPY(MAX_PRNT_SIZE)
	INTEGER*4 NUMPPY(MAX_PRNT_SIZE)
	INTEGER*8 LASTXPRNT(MAX_PRNT_SIZE)
	INTEGER*8 LASTPXPRT(MAX_PRNT_SIZE)
	INTEGER*8 LASTYPRNT(MAX_PRNT_SIZE)

|| INTEGER*8 LASTPYPRNT(MAX_PRNT_SIZE)

Produced using [splitcf](#) by [Dr. K.B.Beard](#)

[splitcf v2.6d8c 9/28/2004 Dr. K.B.Beard]

