FormLink/FeynCalcFormLink Embedding FORM in *Mathematica* and FeynCalc

Feng Feng*[1] and Rolf Mertig†[2]

[1]*Center for High Energy Physics, Peking University, Beijing 100871, China*

[2]*GluonVision GmbH, Bötzowstrasse 10, 10407 Berlin, Germany*

(Dated: December 18, 2012)

Abstract

FORM, a symbolic manipulation system, has been widely used in a lot of calculations for High Energy Physics due to its high performance and efficient design. *Mathematica*, another computational software program, has also widely been used, but more for reasons of generality and user-friendliness than for speed. Especially calculations involving tensors and noncommutative operations like calculating Dirac traces can be rather slow in *Mathematica*, compared to FORM.

In this article we describe FormLink and FeynCalcFormLink, two *Mathematica* packages to link *Mathematica* and FeynCalc with FORM. FormLink can be used without FeynCalc and FeynCalcFormLink, which is an extension loading FormLink and FeynCalc automatically.

With these two packages the impressive speed and other special features of FORM get embedded into the generality of *Mathematica* and FeynCalc in a simple manner.

FeynCalcFormLink provides a FORM-based turbo for FeynCalc, making it much more efficient. FormLink turns *Mathematica* into an editor and code organizer for FORM.

PACS numbers: *12.38.Bx*

*E-mail: fengf@ihep.ac.cn

†E-mail: rolf@mertig.com

PROGRAM SUMMARY

Title of program: `FormLink/FeynCalcFormLink`

Available from: `http://www.feyncalc.org/formlink/`

Programming language: `Mathematica 7, 8 or 9; C; FORM 4`

Computer: Any computer running *Mathematica* and FORM.

Operating system: Linux, Windows, Mac OS X.

External routines: FORM, C, FeynCalc.

Keywords: *Mathematica*, *MathLink*, FORM, FeynCalc, FormLink, FeynCalcFormLink.

Nature of physical problem: The functionality of FORM is restricted compared with *Mathematica*, which is broader, while its speed is slower for larger calculations involving tensors and noncommutative expansions. So how can we combine the speed of FORM with the versatility and broadness of *Mathematica*?

Method of solution: Named pipes, *MathLink*, C, *Mathematica* pre- and postprocessing.

Typical running time: Seconds, minutes, or more, depends on the complexity of the calculation. There is some overhead for conversion of larger expressions.

LONG WRITE-UP

I. Basic Ideas

The basic and simplest way to communicate between *Mathematica* and FORM [1–6] is using input and output files. This method has been used in FeynCalc [7] and FormCalc [8], which prepares the symbolic expressions of the diagrams in an input file for FORM, runs FORM, and retrieves the results back to *Mathematica*.

Another method to exchange data between different processes is to use pipes. The detailed usage of pipe communication between FORM and other external programs is described in [9]. In this article we implement a user-friendly communication between *Mathematica* and FORM using both possibilities. We would like to note that FormCalc and `FormGet.tm`, mentioned at `http://www.feynarts.de/formcalc`, also use *MathLink* and pipes, but do not provide a general interface from *Mathematica* to FORM.

The remainder of this section is intended for programmers only and not needed to understand how to use FormLink.

The basic idea is that we create two unnamed pipes, one is the read-only descriptor with file handler `r#` , the other is the write-only descriptor with file handler `w#`, then start FORM with the `pipe` option automatically through the *MathLink* executable FormLink

```
form -pipe r#,w# init
```

where `init` refers to the FORM code `init.frm` , which is discussed in the following. When the pipe connection has been established successfully, FORM sends its Process Identifer (PID) in ASCII decimal format with an appended newline character to the descriptor `w#` and then FORM will wait for the answer from the descriptor `r#`. The answer must be two comma-separated integers in ASCII decimal format followed by a newline character. The first integer corresponds to the FORM PID while the second one is the PID of parent process which started FORM. If the answer is not obtained after some time-out, or if it is not correct, i.e. it is not a list of two integers or the first integer is not the FORM PID, then FORM fails. When the channel has been established successfully, FORM will run the code in the `init.frm` file, containing the following instructions:

```
Off Statistics;
#ifndef ‘PIPES_’
    #message "No pipes found";
    .end;
#endif
#if (‘PIPES_’ <= 0)
    #message "No pipes found";
    .end;
#endif
#procedure put(fmt, mexp)
    #toexternal ‘fmt’, ‘mexp’
    #toexternal "#THE-END-MARK#"
#endprocedure
#setexternal ‘PIPE1_’;
#toexternal "OK"
```

```
    #fromexternal
    .end
```

The core parts are the last two lines before the `.end` instruction. The first line, `#toexternal "OK"`, sends the word `OK` in ASCII string format from FORM to FormLink, it confirms that the communication channel has been established successfully, otherwise FormLink will treat it as failed. The second line blocks FORM and waits for the code which will be sent from *Mathematica*. When the `prompt`[1] arrives, FORM will continue to execute code from `#fromexternal`. We defined a procedure named `put` to send data from FORM back to *Mathematica*. Note that if you want to send data without this procedure, you need to send the end mark, i.e. the string `#THE-END-MARK#`, to indicate that the data is complete, otherwise FormLink will be blocked until the end mark has been received.

Until now we have demonstrated a round communication from *Mathematica* to FORM, and then back to *Mathematica* again. This procedure can be looped if we put another `#fromexternal` in the code sent from *Mathematica* to FORM, and in this sense, we get an interactive FORM, which cannot be easily achieved by the first way which exchanges data by input and output files.

II. Installation

To install the FormLink and FeynCalcFormLink packages, run the following instruction in a Kernel or Notebook session of *Mathematica* 7, 8 or 9 on Linux, MacOSX, or Windows.

```
    Import["http://www.feyncalc.org/formlink/install.m"]
```

The installer will automatically download `formlink.zip` from the url[2] and extract the files from the archive to the directory `Applications` in the directory `$UserBaseDirectory`, which is by default located in the search path of *Mathematica*. Their values for different platform are listed in TABLE I. Specifying

```
    $installdirectory = mydir
```

before running the installer will change the installation directory. It is recommended to use a directory which is on the *Mathematica* path, e.g., `HomeDirectory[]`, or `$BaseDirectory`.

For user convenience, the binary files of `FormLink`, `form` and `tform` for Linux, Microsoft Windows and Mac OS X are also installed. Furthermore the latest version of FeynCalc is downloaded from `http://www.feyncalc.org` and installed automatically into the same directory unless it has been already installed somewhere on the *Mathematica* path.

Platform	`$UserBaseDirectory`
`Windows`	`C:\Users\`*username*`\AppData\Roaming\Mathematica`
`Linux`	`~/.Mathematica`
`Mac OS X`	`~/Library/Mathematica`

TABLE I: The values of `$UserBaseDirectory` for different operating systems

[1]The default prompt defined in FORM is a blank line, for details please see [9]

[2]`http://www.feyncalc.org/formlink/formlink.zip`

At the end of the installation FormLink and FeynCalcFormLink are loaded and two simple examples are run, one uses FormLink:

```
FormLink[ "    AutoDeclare vector p;
                Local T = g_(0, p1,p2,p3,p4,p5,p6);
                trace4 0;
          "  (*, Form2M -> Identity *)
  (* <-- uncommenting returns a string *)
   ];
```

```
FORM and FormRead finished, time needed before translating to Mathematica: 0. sec
Translation done. Total wall clock time needed: 0.1925 sec
4 p1.p6 p2.p5 p3.p4 – 4 p1.p5 p2.p6 p3.p4 + 4 p1.p2 p3.p4 p5.p6 –
  4 p1.p6 p2.p4 p3.p5 + 4 p1.p4 p2.p6 p3.p5 + 4 p1.p5 p2.p4 p3.p6 – 4 p1.p4 p2.p5 p3.p6 +
  4 p1.p6 p2.p3 p4.p5 – 4 p1.p3 p2.p6 p4.p5 + 4 p1.p2 p3.p6 p4.p5 – 4 p1.p5 p2.p3 p4.p6 +
  4 p1.p3 p2.p5 p4.p6 – 4 p1.p2 p3.p5 p4.p6 + 4 p1.p4 p2.p3 p5.p6 – 4 p1.p3 p2.p4 p5.p6
```

The other example executes FeynCalcFormLink, which generates and runs the corresponding FORM program, substituting ASCII values for greek indices intermediately:

```
FeynCalcFormLink[ DiracTrace[GA[μ, ν, ρ, σ, τ, α]] ]
```

```
AutoDeclare Index lor;
Format Mathematica;
L resFL = (g_(1,lor2)*g_(1,lor3)*g_(1,lor4)*g_(1,lor5)*g_(1,lor6)*g_(1,lor1));
trace4,1;
contract 0;
.sort;
#call put("%E", resFL)
#fromexternal
```

```
Piping the script to FORM and running FORM
Time needed by FORM : 0. seconds. FORM finished. Got the result back to Mathematica as a string.
Start translation to Mathematica / FeynCalc syntax
Total wall clock time used: 0.12 seconds. Translation to Mathematica and FeynCalc finished.
```

$$4 g^{\alpha\tau} g^{\mu\sigma} g^{\nu\rho} - 4 g^{\alpha\sigma} g^{\mu\tau} g^{\nu\rho} + 4 g^{\alpha\mu} g^{\sigma\tau} g^{\nu\rho} - 4 g^{\alpha\tau} g^{\mu\rho} g^{\nu\sigma} + 4 g^{\alpha\rho} g^{\mu\tau} g^{\nu\sigma} + 4 g^{\alpha\sigma} g^{\mu\rho} g^{\nu\tau} - 4 g^{\alpha\rho} g^{\mu\sigma} g^{\nu\tau} + 4 g^{\alpha\tau} g^{\mu\nu} g^{\rho\sigma} - 4 g^{\alpha\nu} g^{\mu\tau} g^{\rho\sigma} + 4 g^{\alpha\mu} g^{\nu\tau} g^{\rho\sigma} - 4 g^{\alpha\sigma} g^{\mu\nu} g^{\rho\tau} + 4 g^{\alpha\nu} g^{\mu\sigma} g^{\rho\tau} - 4 g^{\alpha\mu} g^{\nu\sigma} g^{\rho\tau} + 4 g^{\alpha\rho} g^{\mu\nu} g^{\sigma\tau} - 4 g^{\alpha\nu} g^{\mu\rho} g^{\sigma\tau}$$

The source code of FormLink is included in the src folder. The scripts for compilation are listed in TABLE II. We used the GNU compiler gcc, producing an executable FormLink in the corresponding subdirectory in bin.

Platform	`make` commands

`Microsoft Windows (32 & 64 bit)`	`make -f makefile.cygwin`
	`make -f makefile.cygwin install`

`Linux (32 bit)`	`make -f makefile.linux32`
	`make -f makefile.linux32 install`

`Linux (64 bit)`	`make -f makefile.linux64`
	`make -f makefile.linux64 install`

`Mac OS X (64 bit)`	`make -f makefile.macosx64`
	`make -f makefile.macosx64 install`

TABLE II: The `make` commands for different operating systems. If needed, please modify the corresponding makefile, for example, change the location of the *MathLink* developer kit (`MLINKDIR`) if you did not use the default location during installation of *Mathematica* or if you want to use a different version than *Mathematica* 8.

III. FormLink and FeynCalcFormLink Functions Reference

A. Basic Usage

The basic functions are `FormLink` to execute FORM programs written as a string in *Mathematica* and `FeynCalcFormLink` to run a program specified in FeynCalc syntax through FORM. These two *MathLink*-based functions are easy to use for FORM and FeynCalc practicioners.

- `FormLink`

  `FormLink[" form statements "]` runs the `form statements` in FORM and returns the result to *Mathematica*. If only one `Local` assignment is present the return value is the result. If more FORM `Local` variables are present, a list of results is returned. `FormLink` has several options, the default values are as follows,

  ```
  {Assign → False, Form2M → Form2M, FormSetup :→ $FormSetup, Replace :→ $Form2M,
    Print → True, Style → {RGBColor[0.444444, 0.222222, 0.], FontFamily → Courier}}
  ```

  Here we only explain the most important ones.

  The option `Form2M` can be used to specify the function which translates the expression from FORM to *Mathematica* format, its default value is function `Form2M`, which has the following usage: `Form2M[string, replist]` translates `string` by `ToExpression[StringReplace[string,replist],TraditionalForm]` to *Mathematica*. `Form2M` can also be set to `Identity`, which means that the result from FORM will be returned as a string, containing the exact output for the `Local` expression. If conversion to *Mathematica* is straighforward, i.e., there are no bracketed expressions or other syntaxes not easily interpreted by *Mathematica*, then it is best to use the setting `Form2M→ToExpression`, which will just call `ToExpression` on the string result coming from FORM.

  The option `Replace` is a list of string replacements, its default setting is `$Form2M`, containing a list of user-changeable (e.g. in Config.m) basic function-name translations from FORM to *Mathematica*.

```
{bernoulli_ → BernoulliB, binom_ → Binomial, cos_ → Cos, fac_ → Factorial,
  gcd_ → GCD, ln_ → Log, max_ → Max, min_ → Min, mod_ → Mod, pi_ → Pi,
  li2_ → PolyLog2,, sqrt_ → Sqrt, sign_ → Sign, sin_ → Sin, sum_ → Sum, tan_ → Tan, i_ → I,
  e_ → I*$LeviCivitaSign*Eps, d_ → Pair, \ → , gi_ → DiracGamma,   → , _ → , [ → Hold[Identity][}
```

The option `Print` can be used to switch informative messages on or off during execution. The option `FormSetup` can be a list of FORM settings, like `TempDir` which dynamically produces a `form.set` file next to the binary which is then called automatically upon starting FORM.

- `FeynCalcFormLink`

  `FeynCalcFormLink[expr]` translates the FeynCalc expression `expr` to a FORM program, identifying automatically traces, symbols, vectors, indices and the dimension, calculates it, pipes it back to *Mathematica* and translates it to FeynCalc syntax.

  There are several options for `FeynCalcFormLink`, the default settings are as follows,

```
{Functions → CFunctions, FeynCalcExternal → True, FormSetup :→ $FormSetup,
  Form2FC → Form2FC, ExtraDeclare → {}, IDStatements → {}, Print → True,
  Replace → {}, Style → {RGBColor[0.444444, 0.222222, 0.], FontFamily → Courier}
```

  Most of the options are similar to `FormLink`, we only explain `Functions`, `ExtraDeclare`, `IDStatements` and `Form2FC`.

  If the option `Functions` is set to `‘‘CFunctions’’`, then all non-`System‘` functions, except those present in `$M2Form` and some FeynCalc functions, are automatically declared `CFunctions` in FORM. If `Functions` option is set to `‘‘Functions’’`, then they are declared noncommutative functions, i.e., `Functions` in FORM.

  The option `ExtraDeclare` can be used to put any extra valid FORM declarations which are not identified automatically, for example:

  `ExtraDeclare→{"CFunctions GammaFunction;", "Functions MyOperator;"}`

  The option `IDStatements` can be set to a string or a list of strings corresponding to FORM `identify` statement like `{"id k1.k1=mass^2;"}`.

  The option `Form2FC` is set to the function being used to translate expression from FORM to FeynCalc format, the default is `Form2FC`.

B. Advanced Usage

The basic internal procedure to use FormLink is to start FORM with `FormStart`, then send the code to FORM for execution using `FormWrite` and `FormPrompt`, read and convert the result back to *Mathematica* through `FormRead`, finally, stop FORM by calling `FormStop`. All these calls can be encoded into a single function named `FormLink` which has been introduced in the previous section.

- `FormStart`

  `FormStart[]` automatically determines the path of the `form` executable, which is located in the corresponding subfolder in the `bin` directory, and then starts FORM in pipe mode. You can also call `FormStart[formpath]` with explicit full path `formpath` of the `form` excutable.

- `FormWrite`

  `FormWrite[script]` sends the `script` with an appended newline character, to FORM. Notice that FORM will not start to execute the script right away and you can send your scripts by using the `FormWrite[script]` many times. When you are finished, send the prompt, then FORM executes all instructions sent.

  `FormWrite[scripts]`, where `scripts` is a list of strings, will call `FormWrite[script]` for each element `script` in the list scripts, so if your code spans several lines, you can also put them as a list with each element corresponding to a single line of your scripts.

- `FormPrompt`

  `FormPrompt[]` sends the prompt to FORM, which will make FORM continue to execute the code you have sent. The default prompt in FORM is a blank line, FormLink has adopted this default option, so do not send blank lines unintentionally.

- `FormRead`

  `FormRead[]` reads the data from FORM. It should be noted that if no data is sent from FORM, the calling thread will be blocked until data is available. You can use the procedure `put` defined in `init.frm` to send the data from FORM.

  `FormRead[]` will first check whether the pipe has been closed or not, if the pipe has been closed, for example when FORM has encountered some problems, `FormRead[]` will redirect the standard output from FORM to *Mathematica*, so the user can check the error messages.

- `FormStop`

  `FormStop[]` uninstalls the link to Form. `FormStop[All]` kills all running FormLink processes.

As we discussed in section I, there are two ways to communicate between FORM and *Mathematica*, the functions introduced above are all used with the method of piping. We also provide two functions which are not using *MathLink*, but deal with input and output files only.

- `RunForm`

  `RunForm[script]` runs `script` in FORM and writes the result to `runform.frm` and the log file to `form.log` in the current directory, i.e., the value returned by the *M*athematica function `Directory[]`.

  `RunForm[script, formfile]` uses `formfile` instead of `runform.frm`. The first argument `script` can be a string or a list of strings.

  An optional third argument can be given to use a specific FORM executable, otherwise a FORM executable from `$FormLinkDir/bin` is used.

- `ReadString`

  `ReadString[str]` imports `str` as *Text* and translates it to *Mathematica* syntax by using `$Form2M`. You can use `#write` preprocessor to write your data to the output file by FORM, and use `ReadString` to read it into *Mathematica*.

To facilitate the usage of FormLink with FeynCalc, two functions are provided for performing the conversions between FeynCalc and FORM[3]:

- `FC2Form`

  `FC2Form[exp]` translates `exp` in FeynCalc format to FORM format. `FC2Form[exp]` returns a list of two elements, the first one is a list containing the script which will be sent to FORM. You can use the function `ShowScript[script]` to display the `script`. The second one is a list of replacement rules, which will be used in `Form2FC` to translate the result from FORM format back to FeynCalc. `FC2Form` has the options : `Functions`, `Dimension`, `ExtraDeclare`, `IDStatements`, `Print` and `Replace`:

  ```
  {Functions → CFunctions, Dimension → Automatic,
    ExtraDeclare → {}, IDStatements → {}, Print → True, Replace → {}
  ```

- `Form2FC`

  `Form2FC[formexpr]` is used to translate `formexpr` which is in FORM format back to the FeynCalc one. `Form2FC[formexpr, replacelist]` applies the substitution list `replacelist` at the end. The second argument is usually the second item of of the list returned by `FC2Form`.

So the general steps to use the package with FeynCalc are to convert FeynCalc code to FORM, and then send the converted code to FORM for executing with FormLink, and finally convert the results in FORM format back to FeynCalc. All these steps are implemented into a single function `FeynCalcFormLink`, which has been introduced in the previoius section.

IV. Examples Using FormLink, FeynCalcFormLink and RunForm

We list a few examples using the `FormLink`, `FeynCalcFormLink` and `RunForm` functions. More examples can be found in the `Examples` directories of both packages.

A. Using `FormLink`

`FormLink` is a function for running FORM from *Mathematica* and returning the result to *Mathematica.*

A short trace

```
short = FormLink[ "
                    AutoDeclare vector p;
                    Local T = g_(0, p1,p2,p3,p4,p5,p6);
                    trace4 0;
                "]
```

[3]There are also two older FeynCalc functions named `FeynCalc2FORM` and `FORM2FeynCalc` to perform the conversions.

```
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.001 sec
Translation done. Total wall clock time needed: 0.124 sec
4 p1.p6 p2.p5 p3.p4 - 4 p1.p5 p2.p6 p3.p4 - 4 p1.p6 p2.p4 p3.p5 +
 4 p1.p4 p2.p6 p3.p5 + 4 p1.p5 p2.p4 p3.p6 - 4 p1.p4 p2.p5 p3.p6 +
 4 p1.p6 p2.p3 p4.p5 - 4 p1.p3 p2.p6 p4.p5 + 4 p1.p2 p3.p6 p4.p5 -
 4 p1.p5 p2.p3 p4.p6 + 4 p1.p3 p2.p5 p4.p6 - 4 p1.p2 p3.p5 p4.p6 +
 4 p1.p4 p2.p3 p5.p6 - 4 p1.p3 p2.p4 p5.p6 + 4 p1.p2 p3.p4 p5.p6
% // InputForm
4 * p1 . p6 * p2 . p5 * p3 . p4 -
 4 * p1 . p5 * p2 . p6 * p3 . p4 - 4 * p1 . p6 * p2 . p4 * p3 . p5 +
 4 * p1 . p4 * p2 . p6 * p3 . p5 + 4 * p1 . p5 * p2 . p4 * p3 . p6 -
 4 * p1 . p4 * p2 . p5 * p3 . p6 +
 4 * p1 . p6 * p2 . p3 * p4 . p5 - 4 * p1 . p3 * p2 . p6 * p4 . p5 +
 4 * p1 . p2 * p3 . p6 * p4 . p5 -
 4 * p1 . p5 * p2 . p3 * p4 . p6 + 4 * p1 . p3 * p2 . p5 * p4 . p6 -
 4 * p1 . p2 * p3 . p5 * p4 . p6 +
 4 * p1 . p4 * p2 . p3 * p5 . p6 - 4 * p1 . p3 * p2 . p4 * p5 . p6 +
 4 * p1 . p2 * p3 . p4 * p5 . p6
```

Using the option Form2M→Identity returns a string:

```
short = FormLink[ "
                    AutoDeclare vector p;
                    Local T = g_(0, p1,p2,p3,p4,p5,p6);
                    trace4 0;
                 ", Form2M → Identity
    ]
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.001 sec
Translation done. Total wall clock time needed: 0.1185 sec
(4*p1.p2*p3.p4*p5.p6-4*p1.p2*p3.p5*p4.p6+4*p1.p2*p3.p6*p4.p5-4*p1.p3*p2.p4
  *p5.p6+4*p1.p3*p2.p5*p4.p6-4*p1.p3*p2.p6*p4.p5+4*p1.p4*p2.p3*p5.p6-4*p1.
  p4*p2.p5*p3.p6+4*p1.p4*p2.p6*p3.p5-4*p1.p5*p2.p3*p4.p6+4*p1.p5*p2.p4*p3.
  p6-4*p1.p5*p2.p6*p3.p4+4*p1.p6*p2.p3*p4.p5-4*p1.p6*p2.p4*p3.p5+4*p1.p6*
  p2.p5*p3.p4)
short // StringQ
True
Clear[short]
```

A longer trace

```
medium = FormLink[ "
                    AutoDeclare vector p;
                    Local T = g_(0,
     p1,p2,p3,p4,p5,p6,p7,p8,p9,p10,p11,p12,p13,p3);
                     trace4 0;
                  ",

   Form2M :→ Function[x, ToExpression[StringReplace[x, "." → ""]]]
                  (* Since we want to change p1.p2 to p1p2,
   we replace the . to an empty string "*)
      ];
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.0595 sec
Translation done. Total wall clock time needed: 0.3055 sec
medium // Length
11 763
medium[[1]]
4 p10p13 p11p12 p1p9 p2p8 p3p3 p4p7 p5p6
Variables[medium] // InputForm // Short
{p10p13, p11p12, p1p9, p2p8, p3p3, p4p7,
 p5p6, p10p12, << 67 >>, p9p10, p9p11, p9p12, p9p13}
Clear[medium];
```

Special cases

Objects like `[1-x]` in FORM are translated to `(1-x)` in *Mathematica*:

```
FormLink["
          Symbols y,[1-x];
          L F = [1-x] * (2*y-1)^2*3;
       "]
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.0005 sec
Translation done. Total wall clock time needed: 0.1075 sec
```

$3 (1-x) - 12 (1-x) y + 12 (1-x) y^2$

```
% // Factor
```

$-3 (-1+x) (-1+2 y)^2$

```
FormLink[I]
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.001 sec
Translation done. Total wall clock time needed: 0.0855 sec
```

i

Polynomial examples

Expand $(a+b)^2$

```
FormLink["Symbols a,b; Local F = (a+b)^2"]
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.0005 sec
Translation done. Total wall clock time needed: 0.085 sec
```
$a^2 + 2\,a\,b + b^2$

Alternatively we may just enter the *Mathematica* expression which is then automatically translated to a FORM program:

```
FormLink[(a + b)^2]
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.001 sec
Translation done. Total wall clock time needed: 0.0855 sec
```
$a^2 + 2\,a\,b + b^2$

Expand $(a+b+c+d+e+f+g)^{21}$

The default option setting of `Form2M` does general translation of FORM syntax to *Mathematica*/FeynCalc. Here we can use the simpler `ToExpression` as a setting of `Form2M`. While it is possible to use even 42 instead of 21, we see already for power 21 that it takes longer to transfer the large expression between FORM and *Mathematica* than to calculate it in either FORM or *Mathematica* directly. So it is best not to transfer large expressions, if possible.

```
fresult =
  FormLink["
            Symbols a,b,c,d,e,f,g;
            Local F = (a+b+c+d+e+f+g)^21;
           ",
            Form2M → ToExpression
  ];
FORM and FormRead finished, time
   needed before translating to Mathematica: 1.6787 sec
Translation done. Total wall clock time needed: 5.9193 sec
fresult // Length
296 010
```

So for somewhat larger expressions there is noticable overhead for piping the expression back to *Mathematica* as a string by `FormRead` and for translating the string to *Mathematica* syntax (`ToExpression`).
An accurate timing of FORM can be done by using `RunForm`:

```
RunForm@"   On ShortStats;
            Symbols a,b,c,d,e,f,g;
            Local F = (a+b+c+d+e+f+g)^21;
           ";
FORM 4.0 (Aug 31 × 2012) 32 - bits        Run : Wed Dec 12 × 12 : 01 : 46 × 2012
  On ShortStats;
  Symbols a, b, c, d, e, f, g;
  Local F = (a + b + c + d + e + f + g) ^21;

  .end
   0.24 s    1 >   100 000    100 000 :   3 015 192 F
   0.53 s    1 >   200 000    200 000 :   6 000 862 F
   0.81 s    1 >   296 010    296 010 :   8 643 114 F
   0.85 s    1 >   296 010 -- >   296 010 :   8 643 078 F
 0.85 sec out of 0.85 sec
```

Expand $(a+b+c+d+e+f+g)^{42}$ using `RunForm`

`Last[StringSplit[#,''\n'']]&` is a pure function to extract the last line of the string output of `RunForm`, returning the timing.

```
Last[StringSplit[#, "\n"]] &@
 RunForm["  On ShortStats;
            Symbols a,b,c,d,e,f,g;
            Local F = (a+b+c+d+e+f+g)^42;
           ", Print → False]
  83.85 sec out of 86.46 sec
First @ AbsoluteTiming[ mresult = Expand[(a + b + c + d + e + f + g) ^42];]
63.248618
{Length@ mresult, MemoryInUse[], Clear[mresult], MemoryInUse[]}
{12 271 512, 2 039 817 984, Null, 165 005 184}
```

So `mresult` has more than 12 million terms and it occupies around 2 GB. For this example with a lot of terms FORM 4 is slightly slower than *Mathematica* 9.

```
Clear[fresult, mresult]
```

$(a+b)^6$ and $a^2b \to c$

```
FormLink["
            Symbols a,b,c;
            Local F=(a+b)^6;
            id a^2*b = c;
"]
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.0005 sec
Translation done. Total wall clock time needed: 0.217 sec
```

$a^6 + 6 a b^5 + b^6 + 6 a^3 c + 20 a b^2 c + 15 b^3 c + 15 c^2$

To do the same in *Mathematica* requires more work:

```
Expand[(a + b)^6] //.
 (a^(n_Integer /; n > 1) * (b^(m_.))) → ((a^(n - 2) c) b^(m - 1)))
a^6 + 6 a b^5 + b^6 + 6 a^3 c + 20 a b^2 c + 15 b^3 c + 15 c^2
```

Multiple Local variables: Local F1= $(a+b+c)^{10}$ Local F2 $= (a+b+c+d)^{10}$

```
Length /@ FormLink["
                    #:SmallSize 2000;
                    #:LargePatches 4;
                    Symbols a,b,c,d;
                    Local F1 = (a+b+c)^10;
                    Local F2 = (a+b+c+d)^10;
"]
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.002 sec
Translation done. Total wall clock time needed: 0.1375 sec
{66, 286}
Length /@ Expand[{(a + b + c)^10, (a + b + c + d)^10}]
{66, 286}
```

Commuting and noncommuting functions

```
FormLink["
    Functions A1,B1;
    CFunctions A2,B2;
    Local F1 = (A1+B1)^3;
    Local F2 = (A2+B2)^3;
", Form2M :→ Function[x, ToExpression[StringReplace[x, "*" → "**"]]]
]
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.001 sec
Translation done. Total wall clock time needed: 0.0885 sec
{A1 ** A1 ** A1 + A1 ** A1 ** B1 + A1 ** B1 ** A1 + A1 ** B1 ** B1 + B1 ** A1 ** A1 +
  B1 ** A1 ** B1 + B1 ** B1 ** A1 + B1 ** B1 ** B1, A2^3 + B2^3 + 3 ** A2 ** B2^2 + 3 ** A2^2 ** B2}
```

Index and Vector (Local F=p1(i1)*(p2(i1)+p3(i3))*(p1(i2)+p2(i3));)

```
tmp = FormLink["
               Index i1,i2,i3;
               Vector p1,p2,p3;
               Local F=p1(i1)*(p2(i1)+p3(i3))*(p1(i2)+p2(i3));
"]
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.001 sec
Translation done. Total wall clock time needed: 0.093 sec
```

```
p2.p3 p1[i1] + p1.p2 p1[i2] + p1.p2 p2[i3] + p1[i1] p1[i2] p3[i3]
tmp // InputForm
p2 . p3 * p1[i1] + p1 . p2 * p1[i2] + p1 . p2 * p2[i3] + p1[i1] * p1[i2] * p3[i3]
tmp = FormLink["
                        Index i1,i2,i3;
                        Vector p1,p2,p3;
                        Local F=p1(i1)*(p2(i1)+p3(i3))*(p1(i2)+p2(i3));
", Form2M → Identity]
FORM and FormRead finished, time
   needed before translating to Mathematica: 0.0005 sec
Translation done. Total wall clock time needed: 0.091 sec
(p1(i1)*p1(i2)*p3(i3)+p1(i1)*p2.p3+p1(i2)*p1.p2+p2(i3)*p1.p2)
tmp // InputForm
(p1(i1)*p1(i2)*p3(i3)+p1(i1)*p2.p3+p1(i2)*p1.p2+p2(i3)*p1.p2)
Needs["FeynCalcFormLink`"]
tmp2 =
 Form2FC[tmp, {p1 → Momentum[p1], p2 → Momentum[p2], p3 → Momentum[p3] ,
               i1 → LorentzIndex[i1],
   i2 → LorentzIndex[i2], i3 → LorentzIndex[i3]}]
```

$p1^{i1}\, p1^{i2}\, p3^{i3} + p1^{i1}\, p2 \cdot p3 + p1^{i2}\, p1 \cdot p2 + p2^{i3}\, p1 \cdot p2$

```
tmp2 // InputForm
FV[p1, i1] * FV[p1, i2] * FV[p3, i3] +
 FV[p1, i2] * SP[p1, p2] + FV[p2, i3] * SP[p1, p2] +
 FV[p1, i1] * SP[p2, p3]
tmp2 // FCI // InputForm
Pair[LorentzIndex[i1], Momentum[p1]] *
  Pair[LorentzIndex[i2], Momentum[p1]] *
 Pair[LorentzIndex[i3], Momentum[p3]] +
 Pair[LorentzIndex[i2], Momentum[p1]] *
 Pair[Momentum[p1], Momentum[p2]] + Pair[LorentzIndex[i3], Momentum[p2]] *
 Pair[Momentum[p1], Momentum[p2]] + Pair[LorentzIndex[i1], Momentum[p1]] *
 Pair[Momentum[p2], Momentum[p3]]
Clear[tmp, tmp2]
```

This sets back the default output format type to `StandardForm`.

```
SetOptions[#, "CommonDefaultFormatTypes" →
      {"Input" → StandardForm, "InputInline" → StandardForm,
       "Output" → StandardForm, "OutputInline" → StandardForm,
       "Text" → TextForm, "TextInline" → TraditionalForm}] & /@
  {$FrontEnd, $FrontEndSession};
```

B. Using FeynCalcFormLink

Examples 1

Calculate a trace $\mathtt{tr}(\gamma^\mu\gamma^\nu\gamma^\rho\gamma^\sigma\gamma^\tau\gamma^\sigma)$. Using DiracTrace in FeynCalc does not calculate immediately:

```
spur = DiracTrace[GAD[μ, ν, ρ, σ, τ, σ]]
```
$\mathrm{tr}(\gamma^\mu.\gamma^\nu.\gamma^\rho.\gamma^\sigma.\gamma^\tau.\gamma^\sigma)$

Feeding this into FeynCalcFormLink has enough information to tell FORM to do the trace in D dimensions:

```
FeynCalcFormLink[spur]
```

```
Symbol D;
Dimension D;
AutoDeclare Index lor;
Format Mathematica;
L resFL = (g_(1,lor1)*g_(1,lor2)*g_(1,lor3)*g_(1,lor4)*g_(1,lor5)*g_(1,lor4));
tracen,1;
contract 0;
.sort;
#call put("%E", resFL)
#fromexternal
```

```
Piping the script to FORM and running FORM
Time needed by FORM : 0.006 seconds. FORM finished. Got the result back to Mathematica as a string.
Start translation to Mathematica / FeynCalc syntax
Total wall clock time used: 0.15 seconds. Translation to Mathematica and FeynCalc finished.
```
$-4\,D\,g^{\mu\tau}\,g^{\nu\rho}+4\,D\,g^{\mu\rho}\,g^{\nu\tau}-4\,D\,g^{\mu\nu}\,g^{\rho\tau}+8\,g^{\mu\tau}\,g^{\nu\rho}-8\,g^{\mu\rho}\,g^{\nu\tau}+8\,g^{\mu\nu}\,g^{\rho\tau}$

```
Factor@%
```
$-4\,(D-2)\,(g^{\mu\tau}\,g^{\nu\rho}-g^{\mu\rho}\,g^{\nu\tau}+g^{\mu\nu}\,g^{\rho\tau})$

```
FeynCalcFormLink[DiracTrace[GA[α, β, σ, τ, 5]]]
```

```
AutoDeclare Index lor;
Format Mathematica;
L resFL = (g_(1,lor1)*g_(1,lor2)*g_(1,lor3)*g_(1,lor4)*g5_(1));
trace4,1;
contract 0;
.sort;
#call put("%E", resFL)
#fromexternal
```

```
Piping the script to FORM and running FORM
Time needed by FORM : 0.003 seconds. FORM finished. Got the result back to Mathematica as a string.
```

```
Start translation to Mathematica / FeynCalc syntax
Total wall clock time used: 0.12 seconds. Translation to Mathematica and FeynCalc finished.
```
$4\,\epsilon^{\alpha\beta\sigma\tau}$

Notice that loading FeynCalcFormLink by default sets `$LeviCiviaSign=-I`, such that traces involving γ^5 agree.

```
$LeviCivitaSign
```
$-i$
```
FeynCalcFormLink[
  fun[bla] DiracTrace[somefunction[β] anotherfunction[ϕ] GA[μ, ν, ρ, σ, 7]],
  Print → True] // Factor
```

```
Symbols bla,sym1,sym2;
AutoDeclare Index lor;
AutoDeclare Symbol sym;
CFunctions anotherfunction,fun,somefunction;
Format Mathematica;
L resFL = (anotherfunction(sym2)*fun(bla)*g_(1,lor1)*g_(1,lor2)*g_(1,lor3)*g_(1,lor4)
          *(g7_(1)/2)*somefunction(sym1));
trace4,1;
contract 0;
.sort;
#call put("%E", resFL)
#fromexternal
```

```
Piping the script to FORM and running FORM
Time needed by FORM : 0.003 seconds. FORM finished. Got the result back to Mathematica as a string.
Start translation to Mathematica / FeynCalc syntax
Total wall clock time used: 0.18 seconds. Translation to Mathematica and FeynCalc finished.
```
$-2\,\text{anotherfunction}(\phi)\,\text{fun}(\text{bla})$
$\text{somefunction}(\beta)\,(-g^{\mu\sigma}g^{\nu\rho}+g^{\mu\rho}g^{\nu\sigma}-g^{\mu\nu}g^{\rho\sigma}+\epsilon^{\mu\nu\rho\sigma})$
```
TR[GA[μ, ν, ρ, σ, 7]] // Factor
```
$-2\,(-g^{\mu\sigma}g^{\nu\rho}+g^{\mu\rho}g^{\nu\sigma}-g^{\mu\nu}g^{\rho\sigma}+\epsilon^{\mu\nu\rho\sigma})$

Example 2

```
FeynCalcFormLink[DiracTrace[(m + GS[p]).(M + GS[q])]]
```

```
Symbols m,M;
Vectors p,q;
Format Mathematica;
L resFL = ((m*gi_(1)+g_(1,p))*(M*gi_(1)+g_(1,q)));
trace4,1;
contract 0;
.sort;
#call put("%E", resFL)
```

```
#fromexternal
```

```
Piping the script to FORM and running FORM
Time needed by FORM : 0.002 seconds. FORM finished. Got the result back to Mathematica as a string.
Start translation to Mathematica / FeynCalc syntax
Total wall clock time used: 0.12 seconds. Translation to Mathematica and FeynCalc finished.
```

$4\, m\, M + 4\, p\cdot q$

Example 3

Define simple typesetting rules for p1, ... p8 :

```
Do[With[{v = Symbol["p" <> ToString[i]]},
  MakeBoxes[v, TraditionalForm] = SubscriptBox["p", i]], {i, 8}]
```

Enter a trace $\mathrm{tr}[(\not p_1 + m)\gamma^\mu(\not p_2 + \not k + m)\gamma_\mu \not p_2]$ like this in FeynCalc:

```
exp = DiracTrace[(GS[p1] + m).GAD[μ].(GS[p2 + k] + m).GAD[μ].GS[p2]]
```

$\mathrm{tr}((m + \gamma\cdot \mathrm{p1}).\gamma^\mu.(\gamma\cdot(k + p_2) + m).\gamma^\mu.(\gamma\cdot \mathrm{p2}))$

Calculate it through FORM:

```
R1 = FeynCalcFormLink@exp
```

```
Symbols D,m;
Dimension D;
Vectors k,p1,p2;
AutoDeclare Index lor;
Format Mathematica;
L resFL = ((m*gi_(1)+g_(1,p1))*g_(1,lor1)*(m*gi_(1)+g_(1,k)+g_(1,p2))*g_(1,lor1)
        *g_(1,p2));
tracen,1;
contract 0;
.sort;
#call put("%E", resFL)
#fromexternal
```

```
Piping the script to FORM and running FORM
Time needed by FORM : 0.002 seconds. FORM finished. Got the result back to Mathematica as a string.
Start translation to Mathematica / FeynCalc syntax
Total wall clock time used: 0.13 seconds. Translation to Mathematica and FeynCalc finished.
```

$-4\, D\, m\, k\cdot p_2 - 4\, D\, m\, p_2^2 + 4\, D\, m\, p_1\cdot p_2 + 8\, m\, k\cdot p_2 + 8\, m\, p_2^2$

Check with direct calculation in FeynCalc:

```
R2 = exp /. DiracTrace → Tr
```

$4\left(-D\, m\, k\cdot p_2 - D\, m\, p_2^2 + D\, m\, p_1\cdot p_2 + 2\, m\, k\cdot p_2 + 2\, m\, p_2^2\right)$

```
Expand@FCE[R1 - R2]
```

0

```
Clear[R1, R2]
```

C. Using RunForm

RunForm runs a FORM program without *MathLink*, but by calling FORM directly.

```
Options@RunForm
```

{Style → {RGBColor[0.444444, 0.222222, 0.], FontFamily → Courier},
  FormSetup :→ $FormSetup, Print → True}

```
SetDirectory[$TemporaryDirectory];

AbsoluteTiming[
 RunForm["    symbol a,b,c;
             Format NoSpaces;
             On ShortStats;
             L res= (a+b+c)^5;
             .sort
             #write <res.txt> \"(%E)\", res
             .end
          ",
  "tmp.frm"];
result = Form2M[ReadString["res.txt"]];
 result]
```

```
FORM 4.0 (Aug 31 2012) 32-bits                     Run: Wed Dec 12 13:21:06 2012
        symbol a,b,c;
        Format NoSpaces;
        On ShortStats;
        L res= (a+b+c)^5;
        .sort
        0.00s           1>          21-->          21:          380 res
        #write <res.txt> "(%E)", res
        .end
        0.00s           21>         21-->          21:          380 res
  0.00 sec out of 0.00 sec
```

$\{0.147502,\ a^5 + 5\, a^4\, b + 5\, a^4\, c + 10\, a^3\, b^2 + 20\, a^3\, b\, c + 10\, a^3\, c^2 + 10\, a^2\, b^3 + 30\, a^2\, b^2\, c + 30\, a^2\, b\, c^2 + 10\, a^2\, c^3 + 5\, a\, b^4 + 20\, a\, b^3\, c + 30\, a\, b^2\, c^2 + 20\, a\, b\, c^3 + 5\, a\, c^4 + b^5 + 5\, b^4\, c + 10\, b^3\, c^2 + 10\, b^2\, c^3 + 5\, b\, c^4 + c^5\}$

```
FilePrint["res.txt"]
```

```
(c^5 + 5 * b * c^4 + 10 * b^2 * c^3 + 10 * b^3 * c^2 +
  5 * b^4 * c + b^5 + 5 * a * c^4 + 20 * a * b * c^3 + 30 * a *
   b^2 * c^2 + 20 * a * b^3 * c + 5 * a * b^4 + 10 * a^2 * c^3 +
  30 * a^2 * b * c^2 + 30 * a^2 * b^2 * c +
   10 * a^2 * b^3 + 10 * a^3 * c^2 + 20 * a^3 * b * c +
  10 * a^3 * b^2 + 5 * a^4 * c + 5 * a^4 * b + a^5)
SetDirectory[$TemporaryDirectory]; AbsoluteTiming[

 RunForm[{
   " symbol a,b,c;
       Off Statistics;
       L res= (a+b+c)^199;
      .sort;
      #write <res.txt> \"(%E)\", res
    "
  }, "tmp.frm"];
result = Form2M[ReadString["res.txt"]];
 result // Length]
```

```
FORM 4.0 (Aug 31 2012) 32-bits                    Run: Wed Dec 12 13:21:06 2012
    symbol a,b,c;
    Off Statistics;
    L res= (a+b+c)^199;
    .sort;
    #write <res.txt> "(%E)", res
    .end
  1.21 sec out of 1.22 sec
```

```
{5.324083, 20100}
Length[Expand[(a + b + c)^199] - result]
0
SetDirectory[$TemporaryDirectory]; AbsoluteTiming[
 RunForm[{
   " symbol a,b,c;
       Off Statistics;
       L res= (a+b+c)^199;
       .sort;
     #write <res.txt> \"(%E)\", res
    "
  }, "tmp.frm"];
 (*result = Form2M[ReadString["res.txt"]];*)
 (* if the result from FORM is easy to load into Mathematica,
 Get is faster : *)
result = Get["res.txt"];
 result // Length]
```

```
FORM 4.0 (Aug 31 2012) 32-bits                    Run: Wed Dec 12 13:21:12 2012
```

```
   symbol a,b,c;
   Off Statistics;
   L res= (a+b+c)^199;
   .sort;
   #write <res.txt> "(%E)", res
   .end
  1.21 sec out of 1.21 sec
```

```
{1.605025, 20100}
Length[Expand[(a + b + c)^199] - result]
```

0

For such simple algebraic operations *Mathematica* is actually much faster.

```
AbsoluteTiming[Expand[(a + b + c)^199];]
```

{0.072501, Null}

```
SetDirectory[$TemporaryDirectory];
AbsoluteTiming[
 RunForm[{
    "symbol a,b,c,d,e,f,g;
      Format NoSpaces;
      Off Statistics;
      L res= (a+b+c+d+e+f+g)^21;
     .sort
      #write <res.txt> \"(%E)\", res"
   }, "tmp.frm"];
result = Get["res.txt"]; (*Form2M[ReadString["res.txt"]];*)
 result // Length]
```

```
FORM 4.0 (Aug 31 2012) 32-bits                    Run: Wed Dec 12 13:21:14 2012
   symbol a,b,c,d,e,f,g;
   Format NoSpaces;
   Off Statistics;
   L res= (a+b+c+d+e+f+g)^21;
   .sort
      #write <res.txt> "(%E)", res
   .end
  1.34 sec out of 1.35 sec
```

```
{5.655087, 296010}
AbsoluteTiming[mresult = Expand[(a + b + c + d + e + f + g)^21];]
result - mresult
```

{1.003515, Null}

0

V. APPLICATION TO TREE LEVEL PROCESSES

A. $e^+e^- \to \tau^+\tau^- \to u\,\bar{d}\,\mu\,\bar{\nu}_\mu\,\nu_\tau\,\bar{\nu}_\tau$

Let us consider some applications of FormLink combined with FeynCalc. We take the process: $e^+e^- \to \tau^+\tau^- \to u\,\bar{d}\,\mu\,\bar{\nu}_\mu\,\nu_\tau\,\bar{\nu}_\tau$ as an example, which has been considered in the Form courses[10]. We can express the squared amplitude as:

$$\begin{aligned}
&\frac{1}{2^{16}}\,\mathrm{Tr}\big[(\not{p}_2 - m_e)\gamma^{\mu_1}(\not{p}_1 + m_e)\gamma^{\nu_1}\big]\\
*\;&\mathrm{Tr}\big[(\not{p}_3 + m_3)\gamma^{\mu_2}\gamma_7(\not{q}_1 + m_\tau)\gamma^{\mu_1}(-\not{q}_2 + m_\tau)\gamma^{\mu_3}\gamma_7(\not{p}_6 - m_6)\gamma^{\nu_3}\gamma^7(-\not{q}_2 + m_\tau)\gamma^{\nu_1}(\not{q}_1 + m_\tau)\gamma^{\nu_2}\gamma_7\big]\\
*\;&\mathrm{Tr}\big[(\not{p}_4 + m_4)\gamma^{\mu_2}\gamma_7(\not{p}_5 - m_5)\gamma^{\nu_2}\gamma_7\big]\\
*\;&\mathrm{Tr}\big[(\not{p}_7 + m_7)\gamma^{\mu_3}\gamma_7(\not{p}_8 - m_8)\gamma^{\nu_3}\gamma_7\big] \qquad (1)
\end{aligned}$$

The expression in (1) can be easily translated to FeynCalc syntax and then executed by `FeynCalcFormLink`, it takes a fraction of a second to run the code. The program is:

```
incomingemeppair = DiracTrace[  (GS[p2] - emass).GA[m1].
                                (GS[p1] + emass).GA[n1]];
tauline = DiracTrace[  (GS[p3] + mass3).GA[m2].GA[7].
                       (GS[q1] + tmass).GA[m1].
                       (-GS[q2] + tmass).
                       GA[m3].GA[7].(GS[p6] - mass6).
                       GA[n3].GA[7].(-GS[q2] + tmass).GA[n1].
                      (GS[q1] + tmass).GA[n2].GA[7]
  ];
udbarpair = DiracTrace[
   (GS[p4] + mass4).GA[m2].GA[7].(GS[p5] - mass5).GA[n2].GA[7]];
nubarmupair =
  DiracTrace[(GS[p7] + mass7).GA[m3].GA[7].(GS[p8] - mass8).GA[n3].GA[7]
  ];
```

formexample = $\frac{1}{2^{16}}$2^8 incomingemeppair.tauline.udbarpair.nubarmupair

$$\begin{aligned}
&\frac{1}{256}\,\mathrm{tr}\big((\gamma\cdot \mathrm{p2} - \mathrm{emass}).\gamma^{\mathrm{m1}}.(\mathrm{emass} + \gamma\cdot \mathrm{p1}).\gamma^{\mathrm{n1}}\big).\\
&\quad \mathrm{tr}\big((\mathrm{mass3} + \gamma\cdot \mathrm{p3}).\gamma^{\mathrm{m2}}.\gamma^7.(\gamma\cdot \mathrm{q1} + \mathrm{tmass}).\gamma^{\mathrm{m1}}.(\mathrm{tmass} - \gamma\cdot \mathrm{q2}).\gamma^{\mathrm{m3}}.\\
&\qquad \gamma^7.(\gamma\cdot \mathrm{p6} - \mathrm{mass6}).\gamma^{\mathrm{n3}}.\gamma^7.(\mathrm{tmass} - \gamma\cdot \mathrm{q2}).\gamma^{\mathrm{n1}}.(\gamma\cdot \mathrm{q1} + \mathrm{tmass}).\gamma^{\mathrm{n2}}.\gamma^7\big).\\
&\quad \mathrm{tr}\big((\mathrm{mass4} + \gamma\cdot \mathrm{p4}).\gamma^{\mathrm{m2}}.\gamma^7.(\gamma\cdot \mathrm{p5} - \mathrm{mass5}).\gamma^{\mathrm{n2}}.\gamma^7\big).\mathrm{tr}\big((\mathrm{mass7} + \gamma\cdot \mathrm{p7}).\gamma^{\mathrm{m3}}.\gamma^7.(\gamma\cdot \mathrm{p8} - \mathrm{mass8}).\gamma^{\mathrm{n3}}.\gamma^7\big)
\end{aligned}$$

```
Options[FeynCalcFormLink]
```

{Functions → CFunctions, FeynCalcExternal → True, FormSetup :→ $FormSetup,
  Form2FC → Form2FC, ExtraDeclare → {}, IDStatements → {}, Print → True,
  Replace → {}, Style → {RGBColor[0.444444, 0.222222, 0.], FontFamily → Courier}}

```
R3 = FeynCalcFormLink[formexample, IDStatements → {
      "id q1.q1 = tmass^2;
       id q2.q2 = tmass^2;
       id p1.p2 = s/2-emass^2;
       id q1.q2 = s/2-tmass^2;
    "
    }] // Function[x, Collect2[x, {tmass, emass}]]
```

```
Symbols emass,mass3,mass4,mass5,mass6,mass7,mass8,s,tmass;
Indices m1,m2,m3,n1,n2,n3;
Vectors p1,p2,p3,p4,p5,p6,p7,p8,q1,q2;
Format Mathematica;
L resFL = (((-(emass*gi_(1))+g_(1,p2))*g_(1,m1)*(emass*gi_(1)+g_(1,p1))
          *g_(1,n1)*(mass4*gi_(2)+g_(2,p4))*g_(2,m2)*(g7_(2)/2)*(-(mass5*gi_(2))
          +g_(2,p5))*g_(2,n2)*(g7_(2)/2)*(mass7*gi_(3)+g_(3,p7))*g_(3,m3)
          *(g7_(3)/2)*(-(mass8*gi_(3))+g_(3,p8))*g_(3,n3)*(g7_(3)/2)*(mass3
          *gi_(4)+g_(4,p3))*g_(4,m2)*(g7_(4)/2)*(tmass*gi_(4)+g_(4,q1))
          *g_(4,m1)*(tmass*gi_(4)-g_(4,q2))*g_(4,m3)*(g7_(4)/2)*(-(mass6
          *gi_(4))+g_(4,p6))*g_(4,n3)*(g7_(4)/2)*(tmass*gi_(4)-g_(4,q2))
          *g_(4,n1)*(tmass*gi_(4)+g_(4,q1))*g_(4,n2)*(g7_(4)/2))/256);
trace4,1;
trace4,2;
trace4,3;
trace4,4;
contract 0;
.sort;
id q1.q1 = tmass^2;
id q2.q2 = tmass^2;
id p1.p2 = s/2-emass^2;
id q1.q2 = s/2-tmass^2;
.sort;
#call put("%E", resFL)
#fromexternal
```

```
Piping the script to FORM and running FORM
Time needed by FORM : 0.017 seconds. FORM finished. Got the result back to Mathematica as a string.
Start translation to Mathematica / FeynCalc syntax
Total wall clock time used: 0.2 seconds. Translation to Mathematica and FeynCalc finished.
```

$$\begin{aligned}
&4\,\mathrm{emass}^2\,\mathrm{tmass}^2\,p_3\cdot p_4\,p_6\cdot p_8\\
&\quad(-2\,p_7\cdot \mathrm{q1}\,p_5\cdot \mathrm{q2}-2\,p_5\cdot \mathrm{q1}\,p_7\cdot \mathrm{q2}-2\,p_5\cdot \mathrm{q1}\,p_7\cdot \mathrm{q1}-2\,p_5\cdot \mathrm{q2}\,p_7\cdot \mathrm{q2}+s\,p_5\cdot p_7)+\\
&8\,\mathrm{emass}^2\,s\,p_3\cdot p_4\,p_6\cdot p_8\,p_5\cdot \mathrm{q1}\,p_7\cdot \mathrm{q2}-\\
&2\,\mathrm{tmass}^2\,p_3\cdot p_4\,p_6\cdot p_8\,\big(-2\,s\,p_7\cdot \mathrm{q1}\,p_5\cdot \mathrm{q2}-2\,s\,p_5\cdot \mathrm{q1}\,p_7\cdot \mathrm{q2}-\\
&\qquad 4\,p_5\cdot p_7\,p_2\cdot \mathrm{q1}\,p_1\cdot \mathrm{q2}-4\,p_5\cdot p_7\,p_1\cdot \mathrm{q1}\,p_2\cdot \mathrm{q2}+4\,p_2\cdot p_7\,p_1\cdot \mathrm{q1}\,p_5\cdot \mathrm{q2}+\\
&\qquad 4\,p_1\cdot p_7\,p_2\cdot \mathrm{q1}\,p_5\cdot \mathrm{q2}+4\,p_2\cdot p_5\,p_7\cdot \mathrm{q1}\,p_1\cdot \mathrm{q2}+4\,p_1\cdot p_5\,p_7\cdot \mathrm{q1}\,p_2\cdot \mathrm{q2}+\\
&\qquad 4\,p_2\cdot p_7\,p_1\cdot \mathrm{q1}\,p_5\cdot \mathrm{q1}+4\,p_1\cdot p_7\,p_2\cdot \mathrm{q1}\,p_5\cdot \mathrm{q1}+4\,p_2\cdot p_5\,p_1\cdot \mathrm{q2}\,p_7\cdot \mathrm{q2}+\\
&\qquad 4\,p_1\cdot p_5\,p_2\cdot \mathrm{q2}\,p_7\cdot \mathrm{q2}+s^2\,p_5\cdot p_7-2\,s\,p_1\cdot p_7\,p_2\cdot p_5-2\,s\,p_1\cdot p_5\,p_2\cdot p_7\big)+\\
&16\,p_3\cdot p_4\,p_6\cdot p_8\,p_5\cdot \mathrm{q1}\,p_7\cdot \mathrm{q2}\,(p_2\cdot \mathrm{q1}\,p_1\cdot \mathrm{q2}+p_1\cdot \mathrm{q1}\,p_2\cdot \mathrm{q2})+\\
&4\,s\,\mathrm{tmass}^4\,p_3\cdot p_4\,p_5\cdot p_7\,p_6\cdot p_8
\end{aligned}$$

```
SetSF;
AbsoluteTiming[tmp2 =
  Expand[FeynCalcFormLink[formexample, Print → False] /. {SP[q1, q1] → tmass^2,
     SP[q2, q2] → tmass^2, SP[p1, p2] → s/2 - emass^2, SP[q1, q2] → s/2 - tmass^2}]]
```

```
{0.167021, 4 s tmass^2 SP[p1, p7] SP[p2, p5] SP[p3, p4] SP[p6, p8] +
  4 s tmass^2 SP[p1, p5] SP[p2, p7] SP[p3, p4] SP[p6, p8] +
  4 emass^2 s tmass^2 SP[p3, p4] SP[p5, p7] SP[p6, p8] -
  2 s^2 tmass^2 SP[p3, p4] SP[p5, p7] SP[p6, p8] +
  4 s tmass^4 SP[p3, p4] SP[p5, p7] SP[p6, p8] +
  8 tmass^2 SP[p1, q2] SP[p2, q1] SP[p3, p4] SP[p5, p7] SP[p6, p8] +
  8 tmass^2 SP[p1, q1] SP[p2, q2] SP[p3, p4] SP[p5, p7] SP[p6, p8] -
  8 tmass^2 SP[p1, q1] SP[p2, p7] SP[p3, p4] SP[p5, q1] SP[p6, p8] -
  8 tmass^2 SP[p1, p7] SP[p2, q1] SP[p3, p4] SP[p5, q1] SP[p6, p8] -
  8 tmass^2 SP[p1, q1] SP[p2, p7] SP[p3, p4] SP[p5, q2] SP[p6, p8] -
  8 tmass^2 SP[p1, p7] SP[p2, q1] SP[p3, p4] SP[p5, q2] SP[p6, p8] -
  8 tmass^2 SP[p1, q2] SP[p2, p5] SP[p3, p4] SP[p6, p8] SP[p7, q1] -
  8 tmass^2 SP[p1, p5] SP[p2, q2] SP[p3, p4] SP[p6, p8] SP[p7, q1] -
  8 emass^2 tmass^2 SP[p3, p4] SP[p5, q1] SP[p6, p8] SP[p7, q1] -
  8 emass^2 tmass^2 SP[p3, p4] SP[p5, q2] SP[p6, p8] SP[p7, q1] +
  4 s tmass^2 SP[p3, p4] SP[p5, q2] SP[p6, p8] SP[p7, q1] -
  8 tmass^2 SP[p1, q2] SP[p2, p5] SP[p3, p4] SP[p6, p8] SP[p7, q2] -
  8 tmass^2 SP[p1, p5] SP[p2, q2] SP[p3, p4] SP[p6, p8] SP[p7, q2] +
  8 emass^2 s SP[p3, p4] SP[p5, q1] SP[p6, p8] SP[p7, q2] -
  8 emass^2 tmass^2 SP[p3, p4] SP[p5, q1] SP[p6, p8] SP[p7, q2] +
  4 s tmass^2 SP[p3, p4] SP[p5, q1] SP[p6, p8] SP[p7, q2] +
  16 SP[p1, q2] SP[p2, q1] SP[p3, p4] SP[p5, q1] SP[p6, p8] SP[p7, q2] +
  16 SP[p1, q1] SP[p2, q2] SP[p3, p4] SP[p5, q1] SP[p6, p8] SP[p7, q2] -
  8 emass^2 tmass^2 SP[p3, p4] SP[p5, q2] SP[p6, p8] SP[p7, q2]}
```

```
TraditionalForm[tmp2 // FullSimplify]
```

$-2\, p_3 \cdot p_4\, p_6 \cdot p_8$
$$\Big(\text{tmass}^2 \big(p_5 \cdot p_7 \big(s \big(-2\,\text{emass}^2 + s - 2\,\text{tmass}^2\big) - 4\, p_2 \cdot \text{q1}\, p_1 \cdot \text{q2} - 4\, p_1 \cdot \text{q1}\, p_2 \cdot \text{q2}\big) + 2\, p_7 \cdot \text{q1} \big(2\,\text{emass}^2\, p_5 \cdot \text{q1} + \big(2\,\text{emass}^2 - s\big)\, p_5 \cdot \text{q2} + 2\, p_2 \cdot p_5\, p_1 \cdot \text{q2}\big) + 4\, p_2 \cdot p_7\, p_1 \cdot \text{q1}\, (p_5 \cdot \text{q1} + p_5 \cdot \text{q2})\big) - 2\, p_7 \cdot \text{q2} \big(p_5 \cdot \text{q1} \big(2\,\text{emass}^2 \big(s - \text{tmass}^2\big) + 4\, p_1 \cdot \text{q1}\, p_2 \cdot \text{q2} + s\,\text{tmass}^2\big) - 2\,\text{emass}^2\,\text{tmass}^2\, p_5 \cdot \text{q2} + p_1 \cdot \text{q2} \big(4\, p_2 \cdot \text{q1}\, p_5 \cdot \text{q1} - 2\,\text{tmass}^2\, p_2 \cdot p_5\big)\big) + 2\,\text{tmass}^2\, p_1 \cdot p_7\, (2\, p_2 \cdot \text{q1}\, (p_5 \cdot \text{q1} + p_5 \cdot \text{q2}) - s\, p_2 \cdot p_5) + 2\,\text{tmass}^2\, p_1 \cdot p_5\, (2\, p_2 \cdot \text{q2}\, (p_7 \cdot \text{q1} + p_7 \cdot \text{q2}) - s\, p_2 \cdot p_7)\Big)$$

```
Expand[tmp2 - R3]
0
```

B. Double Bremsstrahlung in τ leptonic radiative decay

This example is provided by Matteo Fael, it refers to a part of the calculation done in [11].

Initialization

```
Up1Up1bar = (*((1 + GA[5] . GS[n])/2) . (GS[p1] + M); *)(GS[p1] + M);
Up2Up2bar = (GS[p2] + m);
```

Bremsstrahlung

Amplitudes without spinors

```
Ma = (Gf/Sqrt[2] * GA[α] . (1 - GA[5])) .
  (I * (GS[p1 - k1 - k2] + M) / (-2 * SP[p1, k1] - 2 * SP[p1, k2] + 2 * SP[k1, k2])) .
  (-I * e * GA[ρ]) . (I * (GS[p1 - k1] + M) / (-2 * SP[p1, k1])) . (-I * e * GA[μ])
```

$$\frac{\gamma^{\alpha}.\left(1-\gamma^{5}\right) G_f}{\sqrt{2}}.\frac{i\,(\gamma \cdot (-k_1 - k_2 + p_1) + M)}{-2\, k_1 \cdot p_1 - 2\, k_2 \cdot p_1 + 2\, k_1 \cdot k_2}.(-i\, e\, \gamma^{\rho}).\left(-\frac{i\,(\gamma \cdot (p_1 - k_1) + M)}{2\, k_1 \cdot p_1}\right).(-i\, e\, \gamma^{\mu})$$

```
Mb = (Gf/Sqrt[2] * GA[α] . (1 - GA[5])) .
  (I * (GS[p1 - k1 - k2] + M) / (-2 * SP[p1, k1] - 2 * SP[p1, k2] + 2 * SP[k1, k2])) .
  (-I * e * GA[μ]) . (I * (GS[p1 - k2] + M) / (-2 * SP[p1, k2])) . (-I * e * GA[ρ])
```

$$\frac{\gamma^{\alpha}.\left(1-\gamma^{5}\right) G_f}{\sqrt{2}}.\frac{i\,(\gamma \cdot (-k_1 - k_2 + p_1) + M)}{-2\, k_1 \cdot p_1 - 2\, k_2 \cdot p_1 + 2\, k_1 \cdot k_2}.(-i\, e\, \gamma^{\mu}).\left(-\frac{i\,(\gamma \cdot (p_1 - k_2) + M)}{2\, k_2 \cdot p_1}\right).(-i\, e\, \gamma^{\rho})$$

```
Mc = (-I * e * GA[ρ]) . (I * (GS[p2 + k2] + m) / (2 * SP[p2, k2])) .
  (Gf/Sqrt[2] * GA[α] . (1 - GA[5])) . (I * (GS[p1 - k1] + M) / (-2 * SP[p1, k1])) . (-I * e * GA[μ])
```

$$(-i\,e\,\gamma^{\rho}).\frac{i\,(\gamma\cdot(k_2+p_2)+m)}{2\,k_2\cdot p_2}.\frac{\gamma^{\alpha}.\left(1-\gamma^{5}\right)G_f}{\sqrt{2}}.\left(-\frac{i\,(\gamma\cdot(p_1-k_1)+M)}{2\,k_1\cdot p_1}\right).(-i\,e\,\gamma^{\mu})$$

```
Md =  (-I * e * GA[μ]) . (I * (GS[p2 + k1] + m)/(2 * SP[p2, k1])) .
   (Gf/Sqrt[2] * GA[α] . (1 - GA[5])) . (I * (GS[p1 - k2] + M)/(-2 * SP[p1, k2])) . (-I * e * GA[ρ])
```

$$(-i\,e\,\gamma^{\mu}).\frac{i\,(\gamma\cdot(k_1+p_2)+m)}{2\,k_1\cdot p_2}.\frac{\gamma^{\alpha}.\left(1-\gamma^{5}\right)G_f}{\sqrt{2}}.\left(-\frac{i\,(\gamma\cdot(p_1-k_2)+M)}{2\,k_2\cdot p_1}\right).(-i\,e\,\gamma^{\rho})$$

```
Me =  (-I * e * GA[ρ]) . (I * (GS[p2 + k2] + m)/(2 * SP[p2, k2])) . (-I * e * GA[μ]) .
   (I * (GS[p2 + k1 + k2] + m) / (2 * SP[p2, k1] + 2 * SP[p2, k2] + 2 * SP[k1, k2])) .
   (Gf/Sqrt[2] * GA[α] . (1 - GA[5]))
```

$$(-i\,e\,\gamma^{\rho}).\frac{i\,(\gamma\cdot(k_2+p_2)+m)}{2\,k_2\cdot p_2}.(-i\,e\,\gamma^{\mu}).\frac{i\,(\gamma\cdot(k_1+k_2+p_2)+m)}{2\,k_1\cdot p_2+2\,k_2\cdot p_2+2\,k_1\cdot k_2}.\frac{\gamma^{\alpha}.\left(1-\gamma^{5}\right)G_f}{\sqrt{2}}$$

```
Mf =  (-I * e * GA[μ]) . (I * (GS[p2 + k1] + m)/(2 * SP[p2, k1])) . (-I * e * GA[ρ]) .
   (I * (GS[p2 + k1 + k2] + m) / (2 * SP[p2, k1] + 2 * SP[p2, k2] + 2 * SP[k1, k2])) .
   (Gf/Sqrt[2] * GA[α] . (1 - GA[5]))
```

$$(-i\,e\,\gamma^{\mu}).\frac{i\,(\gamma\cdot(k_1+p_2)+m)}{2\,k_1\cdot p_2}.(-i\,e\,\gamma^{\rho}).\frac{i\,(\gamma\cdot(k_1+k_2+p_2)+m)}{2\,k_1\cdot p_2+2\,k_2\cdot p_2+2\,k_1\cdot k_2}.\frac{\gamma^{\alpha}.\left(1-\gamma^{5}\right)G_f}{\sqrt{2}}$$

```
Mbrem = Ma + Mb + Mc + Md + Me + Mf;
Mbremstar = FCE[ComplexConjugate[Mbrem] /. {α → β, μ → ν, ρ → σ}];
ComplexConjugate[Ma]
```

$$-\frac{e^2\,G_f\,\gamma^{\mu}.(\gamma\cdot(p_1-k_1)+M).\gamma^{\rho}.(\gamma\cdot(-k_1-k_2+p_1)+M).\left(\gamma^{5}+1\right).\gamma^{\alpha}}{2\sqrt{2}\;k_1\cdot p_1\,(-2\,k_1\cdot p_1-2\,k_2\cdot p_1+2\,k_1\cdot k_2)}$$

Squared Amplitude

ATTENTION! The tensor of Neutrini changes in the double photon emission!

```
Neutriniγγ = 1/(3 * Pi) * ((FV[p1, α] - FV[p2, α] - FV[k1, α] - FV[k2, α]) *
      (FV[p1, β] - FV[p2, β] - FV[k1, β] - FV[k2, β]) -
    MT[α, β] * (M^2 + m^2 - 2 * SP[p1, p2] - 2 * SP[p1, k1] +
        2 * SP[p2, k1] - 2 * SP[p1, k2] + 2 * SP[p2, k2] + 2 * SP[k1, k2]))
```

$$\frac{1}{3\pi}\Big(\left(-k_1{}^{\alpha}-k_2{}^{\alpha}+p_1{}^{\alpha}-p_2{}^{\alpha}\right)\left(-k_1{}^{\beta}-k_2{}^{\beta}+p_1{}^{\beta}-p_2{}^{\beta}\right)-$$
$$g^{\alpha\beta}\left(-2\,k_1\cdot p_1+2\,k_1\cdot p_2-2\,k_2\cdot p_1+2\,k_2\cdot p_2+2\,k_1\cdot k_2+m^2+M^2-2\,p_1\cdot p_2\right)\Big)$$

```
DiracSimplify[DiracGamma[7], DiracSubstitute67 → True]
```

$$\frac{1}{2}-\frac{\gamma^5}{2}$$

```
Fbrem = DiracTrace[MT[μ, ν] MT[ρ, σ] Mbrem.Up1Up1bar.Mbremstar.Up2Up2bar] /.
  {(1 + GA[5]) → (2 GA[6]), (1 - GA[5]) → (2 GA[7]), (1 - GA[6]) → (GA[7])}
```

$$\mathrm{tr}\Bigg(\Bigg((-i\,e\,\gamma^{\mu}).\frac{i\,(m+\gamma\cdot(k_1+p_2))}{2\,k_1\cdot p_2}.(-i\,e\,\gamma^{\rho}).\frac{i\,(m+\gamma\cdot(k_1+k_2+p_2))}{2\,k_1\cdot k_2+2\,k_1\cdot p_2+2\,k_2\cdot p_2}.\frac{\gamma^{\alpha}.(2\,\gamma^7)\,G_f}{\sqrt{2}}+$$
$$(-i\,e\,\gamma^{\mu}).\frac{i\,(m+\gamma\cdot(k_1+p_2))}{2\,k_1\cdot p_2}.\frac{\gamma^{\alpha}.(2\,\gamma^7)\,G_f}{\sqrt{2}}.\left(-\frac{i\,(M+\gamma\cdot(p_1-k_2))}{2\,k_2\cdot p_1}\right).(-i\,e\,\gamma^{\rho})+$$
$$(-i\,e\,\gamma^{\rho}).\frac{i\,(m+\gamma\cdot(k_2+p_2))}{2\,k_2\cdot p_2}.(-i\,e\,\gamma^{\mu}).\frac{i\,(m+\gamma\cdot(k_1+k_2+p_2))}{2\,k_1\cdot k_2+2\,k_1\cdot p_2+2\,k_2\cdot p_2}.\frac{\gamma^{\alpha}.(2\,\gamma^7)\,G_f}{\sqrt{2}}+$$
$$(-i\,e\,\gamma^{\rho}).\frac{i\,(m+\gamma\cdot(k_2+p_2))}{2\,k_2\cdot p_2}.\frac{\gamma^{\alpha}.(2\,\gamma^7)\,G_f}{\sqrt{2}}.\left(-\frac{i\,(M+\gamma\cdot(p_1-k_1))}{2\,k_1\cdot p_1}\right).(-i\,e\,\gamma^{\mu})+$$
$$\frac{\gamma^{\alpha}.(2\,\gamma^7)\,G_f}{\sqrt{2}}.\frac{i\,(M+\gamma\cdot(-k_1-k_2+p_1))}{2\,k_1\cdot k_2-2\,k_1\cdot p_1-2\,k_2\cdot p_1}.(-i\,e\,\gamma^{\mu}).\left(-\frac{i\,(M+\gamma\cdot(p_1-k_2))}{2\,k_2\cdot p_1}\right).(-i\,e\,\gamma^{\rho})+$$
$$\frac{\gamma^{\alpha}.(2\,\gamma^7)\,G_f}{\sqrt{2}}.\frac{i\,(M+\gamma\cdot(-k_1-k_2+p_1))}{2\,k_1\cdot k_2-2\,k_1\cdot p_1-2\,k_2\cdot p_1}.(-i\,e\,\gamma^{\rho}).\left(-\frac{i\,(M+\gamma\cdot(p_1-k_1))}{2\,k_1\cdot p_1}\right).(-i\,e\,\gamma^{\mu})\Bigg).$$
$$(M+\gamma\cdot p_1).\Bigg(-\left(\gamma^{\nu}.(M+\gamma\cdot(p_1-k_1)).\gamma^{\sigma}.(M+\gamma\cdot(-k_1-k_2+p_1)).(2\,\gamma^6).\gamma^{\beta}\,G_f\,e^2\right)\Big/$$
$$\left(2\sqrt{2}\;k_1\cdot p_1\,(2\,k_1\cdot k_2-2\,k_1\cdot p_1-2\,k_2\cdot p_1)\right)-$$
$$\frac{\gamma^{\sigma}.(M+\gamma\cdot(p_1-k_2)).(2\,\gamma^6).\gamma^{\beta}.(m+\gamma\cdot(k_1+p_2)).\gamma^{\nu}\,G_f\,e^2}{4\sqrt{2}\;k_1\cdot p_2\,k_2\cdot p_1}-$$
$$\left(\gamma^{\sigma}.(M+\gamma\cdot(p_1-k_2)).\gamma^{\nu}.(M+\gamma\cdot(-k_1-k_2+p_1)).(2\,\gamma^6).\gamma^{\beta}\,G_f\,e^2\right)\Big/$$
$$\left(2\sqrt{2}\;(2\,k_1\cdot k_2-2\,k_1\cdot p_1-2\,k_2\cdot p_1)\,k_2\cdot p_1\right)-$$
$$\frac{\gamma^{\nu}.(M+\gamma\cdot(p_1-k_1)).(2\,\gamma^6).\gamma^{\beta}.(m+\gamma\cdot(k_2+p_2)).\gamma^{\sigma}\,G_f\,e^2}{4\sqrt{2}\;k_1\cdot p_1\,k_2\cdot p_2}+$$
$$\left((2\,\gamma^6).\gamma^{\beta}.(m+\gamma\cdot(k_1+k_2+p_2)).\gamma^{\sigma}.(m+\gamma\cdot(k_1+p_2)).\gamma^{\nu}\,G_f\,e^2\right)\Big/$$
$$\left(2\sqrt{2}\;k_1\cdot p_2\,(2\,k_1\cdot k_2+2\,k_1\cdot p_2+2\,k_2\cdot p_2)\right)+$$
$$\left((2\,\gamma^6).\gamma^{\beta}.(m+\gamma\cdot(k_1+k_2+p_2)).\gamma^{\nu}.(m+\gamma\cdot(k_2+p_2)).\gamma^{\sigma}\,G_f\,e^2\right)\Big/$$
$$\left(2\sqrt{2}\;k_2\cdot p_2\,(2\,k_1\cdot k_2+2\,k_1\cdot p_2+2\,k_2\cdot p_2)\right)\Bigg).(m+\gamma\cdot p_2)\,g^{\mu\nu}\,g^{\rho\sigma}\Bigg)$$

```
AbsoluteTiming[Amp = FeynCalcFormLink[Neutriniγγ * Fbrem,
    IDStatements →
     "id k1.k1 = 0; id k2.k2 = 0; id p1.p1 = M^2; id p2.p2 = m^2;"
   ];]
```

```
Symbols e,Gsubf,m,M;
Vectors k1,k2,p1,p2;
AutoDeclare Index lor;
Format Mathematica;
L resFL = ((d_(lor3,lor4)*d_(lor5,lor6)*(-((e^2*Gsubf*g_(1,lor1)*(g7_(1)/2)*(M-g_(1,k1)
        -g_(1,k2)+g_(1,p1))*g_(1,lor5)*(M-g_(1,k1)+g_(1,p1))*g_(1,lor3))/(sqrt_(2)
        *p1(k1)*(2*k2(k1)-2*p1(k1)-2*p1(k2))))-(e^2*Gsubf*g_(1,lor1)*(g7_(1)/2)
        *(M-g_(1,k1)-g_(1,k2)+g_(1,p1))*g_(1,lor3)*(M-g_(1,k2)+g_(1,p1))
        *g_(1,lor5))/(sqrt_(2)*(2*k2(k1)-2*p1(k1)-2*p1(k2))*p1(k2))-(e^2*Gsubf
        *g_(1,lor3)*(m+g_(1,k1)+g_(1,p2))*g_(1,lor1)*(g7_(1)/2)*(M-g_(1,k2)
        +g_(1,p1))*g_(1,lor5))/(2*sqrt_(2)*p1(k2)*p2(k1))-(e^2*Gsubf*g_(1,lor5)
        *(m+g_(1,k2)+g_(1,p2))*g_(1,lor1)*(g7_(1)/2)*(M-g_(1,k1)+g_(1,p1))
        *g_(1,lor3))/(2*sqrt_(2)*p1(k1)*p2(k2))+(e^2*Gsubf*g_(1,lor3)*(m+g_(1,k1)
        +g_(1,p2))*g_(1,lor5)*(m+g_(1,k1)+g_(1,k2)+g_(1,p2))*g_(1,lor1)
        *(g7_(1)/2))/(sqrt_(2)*p2(k1)*(2*k2(k1)+2*p2(k1)+2*p2(k2)))+(e^2*Gsubf
        *g_(1,lor5)*(m+g_(1,k2)+g_(1,p2))*g_(1,lor3)*(m+g_(1,k1)+g_(1,k2)+g_(1,p2))
        *g_(1,lor1)*(g7_(1)/2))/(sqrt_(2)*p2(k2)*(2*k2(k1)+2*p2(k1)+2*p2(k2))))
        *(M*gi_(1)+g_(1,p1))*(-((e^2*Gsubf*g_(1,lor4)*(M-g_(1,k1)+g_(1,p1))
        *g_(1,lor6)*(M-g_(1,k1)-g_(1,k2)+g_(1,p1))*(g6_(1)/2)*g_(1,lor2))/(sqrt_(2)
        *p1(k1)*(2*k2(k1)-2*p1(k1)-2*p1(k2))))-(e^2*Gsubf*g_(1,lor6)*(M-g_(1,k2)
        +g_(1,p1))*g_(1,lor4)*(M-g_(1,k1)-g_(1,k2)+g_(1,p1))*(g6_(1)/2)
        *g_(1,lor2))/(sqrt_(2)*(2*k2(k1)-2*p1(k1)-2*p1(k2))*p1(k2))-(e^2*Gsubf*g_(1,lor6)
        *(M-g_(1,k2)+g_(1,p1))*(g6_(1)/2)*g_(1,lor2)*(m+g_(1,k1)+g_(1,p2))
        *g_(1,lor4))/(2*sqrt_(2)*p1(k2)*p2(k1))-(e^2*Gsubf*g_(1,lor4)*(M-g_(1,k1)
        +g_(1,p1))*(g6_(1)/2)*g_(1,lor2)*(m+g_(1,k2)+g_(1,p2))*g_(1,lor6))/(2
        *sqrt_(2)*p1(k1)*p2(k2))+(e^2*Gsubf*(g6_(1)/2)*g_(1,lor2)
        *(m+g_(1,k1)+g_(1,k2)+g_(1,p2))*g_(1,lor6)*(m+g_(1,k1)+g_(1,p2))
        *g_(1,lor4))/(sqrt_(2)*p2(k1)*(2*k2(k1)+2*p2(k1)+2*p2(k2)))+(e^2*Gsubf
        *(g6_(1)/2)*g_(1,lor2)*(m+g_(1,k1)+g_(1,k2)+g_(1,p2))*g_(1,lor4)
        *(m+g_(1,k2)+g_(1,p2))*g_(1,lor6))/(sqrt_(2)*p2(k2)*(2*k2(k1)+2*p2(k1)
        +2*p2(k2))))*(m*gi_(1)+g_(1,p2))*((-k1(lor1)-k2(lor1)+p1(lor1)-p2(lor1))*(-k1(lor2)
        -k2(lor2)+p1(lor2)-p2(lor2))-d_(lor1,lor2)*(m^2+M^2+2*k2(k1)
        -2*p1(k1)-2*p1(k2)+2*p2(k1)+2*p2(k2)-2*p2(p1))))/(3*pi_));
trace4,1;
contract 0;
.sort;
id k1.k1 = 0;
id k2.k2 = 0;
id p1.p1 = M^2;
id p2.p2 = m^2;
.sort;
#call put("%E", resFL)
#fromexternal
```

Piping the script to FORM and running FORM

Time needed by FORM : 4.583 seconds. FORM finished. Got the result back to Mathematica as a string.

Start translation to Mathematica / FeynCalc syntax

Total wall clock time used: 13.28 seconds. Translation to Mathematica and FeynCalc finished.

```
{13.378199, Null}
Variables[Amp]
```

$\{e, m, M, k_1\cdot k_2, k_1\cdot p_1, k_1\cdot p_2, k_2\cdot p_1, k_2\cdot p_2, p_1\cdot p_2, G_f\}$

```
% // InputForm
{e, m, M, SP[k1, k2], SP[k1, p1], SP[k1, p2],
 SP[k2, p1], SP[k2, p2], SP[p1, p2], Subscript[G, f]}
FORMAmp = Expand[Amp];
Length@FORMAmp
```

2612

Check with FeynCalc

```
ScalarProduct[k1, k1] = 0;
ScalarProduct[k2, k2] = 0;
ScalarProduct[p1, p1] = M^2;
ScalarProduct[p2, p2] = m^2;
```

The naive approach is too slow.

```
(*AbsoluteTiming[FCAmp = TR[Neutriniγγ * Fbrem];]*)
```

Do this instead (also still quite slow, but good enough for checking equality)

```
AbsoluteTiming[FCAmp1 = Contract[Neutriniγγ DiracGammaExpand[FCI[Fbrem]]] /.
    DiracTrace → DiracTrick; DotExpand2[expr_] :=
  expr /. Dot → Hold[Dot] //. {Hold[Dot][a___, b_Plus, c___] :→
      (Distribute[Hold[Dot][a, b, c]] //. {Hold[Dot][aa___, bb_ cc_, dd___] :→
          bb Hold[Dot][aa, cc, dd] /; NonCommFreeQ[bb], Hold[Dot][aa___,
           bb_, dd___] :→ bb Hold[Dot][aa, dd] /; NonCommFreeQ[bb]})} /.
    {Hold[Dot][] :→ 1, Hold[Dot] :→ Dot}; Print[
  "noncommutative expansion, time needed: ",
  AbsoluteTiming[
    FCAmp2 = Expand[DotExpand2[FCAmp1], Dot];]];
 Print["collecting traces , time needed: ",
  AbsoluteTiming[
    FCAmp3 = Collect2[DiracTrick[FCAmp2], Dot, Factoring → False];]];
 Print["Length[FCAmp3] = ", Length[FCAmp3]];
 Print["time doing the traces ",
  AbsoluteTiming[zeit = AbsoluteTime[]; FCAmp4 = Sum[(If[IntegerQ[i/50],
        PrintTemporary[{i, "  ", Round[AbsoluteTime[] - zeit]}]];
       Expand[ExpandScalarProduct@TR[FCAmp3[[i]]]]), {i, Length[FCAmp3]}];]];
 FCAmp5 = Expand[ScalarProductExpand[FCAmp4]];
 Print["length of FCAmp5 = ", Length[FCAmp5]]]
```

noncommutative expansion, time needed: {44.751183, Null}

```
collecting traces , time needed: {169.767466, Null}
Length[FCAmp3] = 694
time doing the traces {62.221846, Null}
length of FCAmp5 = 1896
{277.372532, Null}
AbsoluteTiming[diff = Collect[FCE[FCAmp5 - Amp], M, Factor]]
{16.162748, 0}
```

The results of FORM and FeynCalc agree, but FORM is much faster. There is a some noticeable overhead in translating the returned string to FeynCalc. In a future version of FormLink we will improve this by either using parallel features of *Mathematica* or by writing a special *MathLink* program similar to `FormGet.tm`.

VI. Summary

We have implemented FormLink and FeynCalcFormLink, two *MathLink*-based programs implemented in *Mathematica*, C and FORM, which embed FORM in *Mathematica* and FeynCalc. A non-*MathLink*-based file-based function RunForm has been also put into the FormLink package. For both packages we provide configurable functions for the syntax conversions. A limited subset of the syntaxes from both programs are automatically translated to each other by our two *Mathematica* packages. In this way we can combine the speed of FORM with the generality of *Mathematica*. A simple installation facility has been provided.

We hope to be able to extend the functionality and range of applicability of the packages in the future.

VII. Licenses

FormLink and FeynCalcFormLink are covered by the GNU Lesser General Public License, like FORM. The conditions for the use of FORM are laid out here: `http://www.nikhef.nl/~form/license/license.html` and should be followed of course also when using FormLink/FeynCalcFormLink. The license for using *Mathematica* is given here: `http://www.wolfram.com/legal/agreements/wolfram-mathematica.html`

Acknowledgments

We would like to thank Matteo Fael for providing us with the example about double bremsstrahlung in τ leptonic radiative decay. Rolf Mertig would like to thank Prof Dr Thomas Gehrmann from the Institut für Theoretische Physik, Universität Zürich, for the invitation to teach *Mathematica* for High Energy Physics students, which encouraged him to work on `FormLink/FeynCalcFormLink`.

Feng Feng wants to thank Hai-Rong Dong, Wen-Long Sang and Prof. Yu Jia for many useful discussions. Finally, Feng Feng would like to commemorate his beloved mother.

[1] J. Kuipers, T. Ueda, J. A. M. Vermaseren and J. Vollinga, FORM version 4.0, arXiv:1203.6543 [cs.SC].

[2] J. A. M. Vermaseren, math-ph/0010025.

[3] J. A. M. Vermaseren, FORM development, PoS CPP **2010**, 012 (2010) [arXiv:1101.0511 [hep-ph]].

[4] J. A. M. Vermaseren, FORM facts, Nucl. Phys. Proc. Suppl. **205-206**, 104 (2010) [arXiv:1006.4512 [hep-ph]].

[5] M. Tentyukov and J. A. M. Vermaseren, Current status of FORM parallelization, PoS ACAT **08** (2008) 119.

[6] J. A. M. Vermaseren, The FORM project, Nucl. Phys. Proc. Suppl. **183**, 19 (2008) [arXiv:0806.4080 [hep-ph]].

[7] R. Mertig, M. Bohm and A. Denner, FEYNCALC: Computer algebraic calculation of Feynman amplitudes, Comput. Phys. Commun. **64** (1991) 345.

[8] T. Hahn and M. Perez-Victoria, Automatized one loop calculations in four-dimensions and D-dimensions, Comput. Phys. Commun. **118**, 153 (1999) [hep-ph/9807565].

[9] M. Tentyukov and J. A. M. Vermaseren, Extension of the functionality of the symbolic program FORM by external software, Comput. Phys. Commun. **176**, 385 (2007) [cs/0604052 [cs-sc]].

[10] Jos Vermaseren, Introduction to FORM,
`http://www.nikhef.nl/~form/maindir/courses/course1/sheets5.pdf`

[11] A. Fischer, T. Kurosu and F. Savatier, QED one loop correction to radiative muon decay, Phys. Rev. D **49**, 3426 (1994).