You can check the status of a SPID using any of these queries:

SELECT status,\* FROM sys.sysprocesses

 SELECT status,\* FROM sys.dm\_exec\_requests

SELECT status,\* FROM sys.dm\_exec\_sessions

EXEC sp\_who2

It should be noted that all these queries may not return the same amount of results. For ex: dm\_exec\_requests DMV returns information about requests that are currently executing (RUNNABLE and RUNNING) within SQL Server, so any request that is not currently executing may not showup in the resultset from this DMV.

Before we get into the description about SPID status, let's understand details about SCHEDULER in SQL Server Operating System(SQLOS):

There will be one scheduler (Called SQLOS SCHEDULER) in SQL Server each mapped to a logical processor exposed to SQL Server. Wondering what about hyperthreading enabled machines? Refer the line above which says "logical processor" which means we already have 2 schedulers for 1 physical processor with hyperthreading enabled. Each scheduler can only have one active, running, task. The rest of the tasks that are ready to run are not running but rather runable and located in scheduler runable queue. So at any given time, each scheduler will have at most a single running user, a runnable queue of requests that are waiting for CPU, a waiter list (for resources such as IO, locks, memory), and a work queue (user requests that are waiting for worker threads). You can find the number of schedulers created in your SQL Server using the query: SELECT \* FROM sys.dm\_os\_schedulers WHERE STATUS = 'VISIBLE ONLINE'

Now let's focus on some of the most seen status of SPID's in SQL Server and what do they mean:

**RUNNING:**This status means session is running one or more batches. When Multiple Active Result Sets (MARS) is enabled, a session can run multiple batches. (Source: BOL)  
What this actually means is, the client connected to SQL Server using this session has already submitted a query for SQL Server to process and SQL Server is currently processing the query. The query could be anywhere between generating a parser tree to performing a join to sorting the data... and it is consuming the CPU (Processor) cycles currently.

**SUSPENDED:**It means that the request currently is not active because it is waiting on a resource. The resource can be an I/O for reading a page, A WAITit can be communication on the network, or it is waiting for lock or a latch. It will become active once the task it is waiting for is completed. For example, if the query the has posted a I/O request to read data of a complete table tblStudents then this task will be suspended till the I/O is complete. Once I/O is completed (Data for table tblStudents is available in the memory), query will move into RUNNABLE queue.

So if it is waiting, check the wait\_type column to understand what it is waiting for and troubleshoot based on the wait\_time. "SQL Server 2005 Waits and Queues" whitepaper published by SQL Cat team will be a good read to troubleshoot performance issue based on Waits. Download it from[**http://sqlcat.com/whitepapers/archive/2007/11/19/sql-server-2005-waits-and-queues.aspx**](http://sqlcat.com/whitepapers/archive/2007/11/19/sql-server-2005-waits-and-queues.aspx)

You can also check my blog [**http://blogs.msdn.com/b/sqlsakthi/archive/2011/02/20/sql-query-slowness-troubleshooting-using-extended-events-wait-info-event.aspx**](http://blogs.msdn.com/b/sqlsakthi/archive/2011/02/20/sql-query-slowness-troubleshooting-using-extended-events-wait-info-event.aspx) to find out waits for a particular SPID using XEVENTS.

Here is a query that returns information about the wait queue of tasks that are waiting on some resource:

SELECT  wt.session\_id, ot.task\_state, wt.wait\_type, wt.wait\_duration\_ms, wt.blocking\_session\_id, wt.resource\_description, es.[host\_name], es.[program\_name] FROM  sys.dm\_os\_waiting\_tasks  wt  INNER  JOIN sys.dm\_os\_tasks ot ON ot.task\_address =wt.waiting\_task\_address INNER JOIN sys.dm\_exec\_sessions es ON es.session\_id = wt.session\_id WHERE es.is\_user\_process =  1

**RUNNABLE:**The SPID is in the runnable queue of a scheduler and waiting for a quantum to run on the scheduler. This means that requests got a worker thread assigned but they are not getting CPU time.

The RUNNABLE queue can be likened to a grocery analogy where there are multiple check out lines.  The register clerk is the CPU.  There is just one customer checking out e.g. “RUNNING” at any given register.  The time spent in the checkout line represents CPU pressure. So this SPID is waiting for that customer who is running (with register clerk) to get out so that it can start RUNNING.

Now, we cannot say that the system does not have enough CPU if you see more SPID's with status RUNNABLE. Your load is really CPU bounded if a number of runnable tasks per each scheduler always greater than 1 and all of your queries have correct plan. So you have to make sure that plan generated is the effective plan but still is the query is forced to wait in RUNNABLE queue for a longer time, then adding more CPU makes sense.

How would you ensure that plan generated is the effective one?   
Good question. Right?  
Simple answer to this would be:  
1. Statistics are up to date? (Including manually created, auto-created and stats created by indexes)  
2. Proper MAXDOP settings (Refer KB 329204, 2023536) **etc..**   
Since this is deviating from the topic of the blog, I'm stopping the discussion about why we see more SPID's in RUNNABLE queue here.

You can use the query SELECT wait\_type,waiting\_tasks\_count,signal\_wait\_time\_ms FROM sys.dm\_os\_wait\_stats ORDER BY signal\_wait\_time\_ms DESC  to find out the difference between the time the waiting thread was signaled and when it started running. This difference is the time spent in RUNNABLE queue. Some of the waits on the top of the list can be safely ignored. Wait\_types with usage "Background" as specified in[**http://blogs.msdn.com/b/psssql/archive/2009/11/03/the-sql-server-wait-type-repository.aspx**](http://blogs.msdn.com/b/psssql/archive/2009/11/03/the-sql-server-wait-type-repository.aspx) are the ones to be ignored from the output of this DMV.

**PENDING**:  
The request is waiting for a worker to pick it up.  
This means the request is ready to run but there are no worker threads available to execute the requests in CPU.  This doesn't mean that you have to increase 'Max. Worker threads", you have to check what the currently executing threads are doing and why they are not yielding back. I personally have seen more SPID's with status PENDING on issues which ended up in "Non-yielding Scheduler" and "Scheduler deadlock". Check Q4 in Slava Oak's blog[**http://blogs.msdn.com/b/slavao/archive/2006/09/28/776437.aspx**](http://blogs.msdn.com/b/slavao/archive/2006/09/28/776437.aspx) to decide on when to increase Max. worker threads.

**BACKGROUND**:   
The request is a background thread such as Resource Monitor or Deadlock Monitor.

**SLEEPING**:  
There is no work to be done.

I will keep adding additional information as and when I have it.